entropy

MDPI

Article

F-Divergences and Cost Function Locality in Generative Modelling with Quantum Circuits

Chiara Leadbeater [1,†], Louis Sharrock [1,2,†], Brian Coyle [1,3] and Marcello Benedetti [1,*]

[1] Cambridge Quantum Computing Limited, London SW1E 6DR, UK; chiara.leadbeater@cambridgequantum.com (C.L.); louis.sharrock@cambridgequantum.com (L.S.); brian.coyle@cambridgequantum.com (B.C.)
[2] Department of Mathematics, Imperial College London, London SW7 2AZ, UK
[3] School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, UK
[*] Correspondence: marcello.benedetti@cambridgequantum.com
[†] These authors contributed equally to this work.

Abstract: Generative modelling is an important unsupervised task in machine learning. In this work, we study a hybrid quantum-classical approach to this task, based on the use of a quantum circuit born machine. In particular, we consider training a quantum circuit born machine using f-divergences. We first discuss the adversarial framework for generative modelling, which enables the estimation of any f-divergence in the near term. Based on this capability, we introduce two heuristics which demonstrably improve the training of the born machine. The first is based on f-divergence switching during training. The second introduces locality to the divergence, a strategy which has proved important in similar applications in terms of mitigating barren plateaus. Finally, we discuss the long-term implications of quantum devices for computing f-divergences, including algorithms which provide quadratic speedups to their estimation. In particular, we generalise existing algorithms for estimating the Kullback–Leibler divergence and the total variation distance to obtain a fault-tolerant quantum algorithm for estimating another f-divergence, namely, the Pearson divergence.

Keywords: generative modelling; born machine; f-divergence; local cost function

check for updates

Citation: Leadbeater, C.; Sharrock, L.; Coyle, B.; Benedetti, M. F-Divergences and Cost Function Locality in Generative Modelling with Quantum Circuits. *Entropy* **2021**, *23*, 1281. https://doi.org/10.3390/e23101281

Academic Editors: Diego Oliva and Salvador Miguel Hinojosa Cervantes

Received: 7 September 2021
Accepted: 28 September 2021
Published: 30 September 2021

1. Introduction

One of the most challenging technological questions of our time is whether existing quantum computers can achieve quantum advantage in tasks of practical interest. Variational quantum algorithms (VQAs), which are well suited to the constraints imposed by existing devices, have emerged as the leading strategy for achieving such a quantum advantage [1–4].

In VQAs, a problem-specific cost function, which typically consists of a functional of the output of a parameterised quantum circuit, is efficiently evaluated using a quantum computer. Meanwhile, a classical optimiser is leveraged to train the circuit parameters in order to minimise the cost function. This hybrid quantum-classical approach is robust to the limited connectivity and qubit count of existing devices, and, by restricting the circuit depth, also provides an effective strategy for error mitigation.

Given their flexibility, VQAs have been proposed for a vast array of applications. Of particular relevance are applications of VQAs to machine learning problems, including classification [5–10], data compression [11–13], clustering [14], generative modelling [15–32] , and inference [33].

In this paper, we focus on a hybrid quantum-classical approach to generative modelling using a born machine [34]. We adopt an adversarial framework to this task, in which a born machine (the 'generator') generates samples from the target distribution, while a binary classifier (the 'discriminator') attempts to distinguish between generated samples and true samples. This is sometimes referred to in the literature as a quantum generative adversarial network.

In a generalisation of existing approaches, we consider training the born machine with respect to any f-divergence as a cost function. Well-known examples of f-divergences include the Kullback–Leibler divergence (KL), the Jensen–Shannon divergence (JS), the squared Hellinger distance (H^2), the total variation distance (TV), and the Pearson divergence (χ^2). In the adversarial framework, it is straightforward to estimate the f-divergence: any such divergence is defined in terms of the density ratio of the target distribution and model distribution, which can be estimated using standard techniques via the output of the binary classifier [35]. On this basis, we propose a heuristic for training the born machine, based on the idea of dynamically switching the f-divergence during training in order to optimise the rate of convergence and utilise favourable qualities of each one. We also propose a second heuristic, based on introducing locality into the f-divergence, motivated by the now well-established connection between locality and barren plateaus in VQA training landscapes [36,37]. For both heuristics, we provide numerical evidence to suggest that they can lead to (sometimes significant) performance improvements, particularly in under- and over-parameterised circuits.

We conclude this paper with a discussion of the longer-term implication of quantum devices for computing the f-divergences between two probability distributions. In particular, we discuss the existence of quadratic speedups for the estimation of TV and KL shown by [38–40] and extend these results to an algorithm for estimating χ^2, assuming access to a fault-tolerant quantum computer.

The remainder of this paper is organised as follows. In Section 2, we begin by introducing generative modelling, Born machines, and f-divergences. In Section 3, we then introduce the two training heuristics for the born machine. In Section 4, we provide numerical results to demonstrate the performance of the heuristics. In Section 5, we discuss the long-term implications of quantum devices for computing f-divergences. Finally, in Section 6, we offer some concluding remarks.

2. Background

2.1. Generative Modelling

Generative modelling is an unsupervised machine learning task in which the goal is to learn the probability distribution which generates a given data set. More precisely, given access to i.i.d. samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x})$ in $\mathbb{R}^p$, the objective of generative modelling is to learn a model $q_{\boldsymbol{\theta}}(\boldsymbol{x})$, typically parameterised by a d dimensional parameter vector, $\boldsymbol{\theta} \in \mathbb{R}^d$, which closely resembles $p(\boldsymbol{x})$. Generative models find applications in a wide range of problems, ranging from the typical modalities of machine learning such as text [41], image [42] and graph [43] analysis, to problems in active learning [44], reinforcement learning [45], medical imaging [46], physics [47], and speech synthesis [48].

Broadly speaking, one can distinguish between two main categories of generative model: prescribed models and implicit models [49,50]. Prescribed models provide an explicit parametric specification of the distribution of the observed random variable $\boldsymbol{x}$, directly specifying the density $q_{\boldsymbol{\theta}}(\boldsymbol{x})$. An example of a prescribed model is the ubiquitous multivariate Gaussian distribution. Implicit models, on the other hand, specify only the stochastic procedure which generates samples. An example of an implicit model is a complex computer simulation of some physical phenomenon, for which the likelihood function cannot be computed. Since, in this case, one no longer models $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ directly, valid objectives can now only involve quantities (e.g., expectation values) which can be estimated efficiently using samples.

In the last three decades, a number of generative models, both explicit and implicit, have been proposed in the machine learning literature. These include autoregressive models [51,52], normalising flows [53–55], variational autoencoders [56,57], Boltzmann machines [58–60], generative stochastic networks [61], generative moment matching networks [62,63], and generative adversarial networks [64]. These models are classically implemented using deep neural network architectures. In recent years, however, hybrid quantum-classical approaches based on parameterised quantum circuits have also gained traction [15–32].

2.2. Born Machines as Implicit Generative Models

By directly exploiting born's probabilistic interpretation of quantum wave functions [65], it is possible to model the probability distribution of classical data using a pure quantum state. Such models are referred to as born machines [34]. We are particularly interested in born machines for which the quantum state is obtained via a parameterised quantum circuit (as opposed to, say, a continuous time Hamiltonian evolution). These are known as quantum circuit born machines (QCBMs) [15,16].

The use of QCBMs as generative models is in large part motivated by their expressiveness. Indeed, it is now well established that born machines have greater expressive power than classical models, including neural networks [20] and partially matrix product states [66] (see also [19]). This means, in particular, that QCBMs can efficiently represent certain distributions which are classically intractable to simulate (e.g., [67–69]). These include those recently used in a demonstration of quantum supremacy [70].

Let us consider a binary vector $\boldsymbol{x} \in \{0,1\}^n$, with n the number of qubits. A QCBM takes a product state $|0\rangle^{\otimes n}$ as input and evolves it into a normalised output state $|\Psi(\boldsymbol{\theta})\rangle$ via a parameterised quantum circuit $U(\boldsymbol{\theta})$. One can generate n-bit strings according to

$$\boldsymbol{x} \sim q_{\boldsymbol{\theta}}(\boldsymbol{x}) = |\langle \boldsymbol{x}|\Psi(\boldsymbol{\theta})\rangle|^2, \tag{1}$$

where $|\boldsymbol{x}\rangle$ are computational basis states; sampling from this distribution then consists of a simple measurement. Since we only have access to $\boldsymbol{x} \sim q_{\boldsymbol{\theta}}(\boldsymbol{x})$ and not the probabilities, $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ themselves, the born machine can be regarded as an implicit generative model. We consider parameterised quantum circuits $U(\boldsymbol{\theta})$ of the form

$$U(\boldsymbol{\theta}) = \prod_{i=1}^{D} W_i U_i(\theta_i), \tag{2}$$

where $\{W_i\}_{i=1}^D$ is a set of fixed unitaries, $\{U_i(\theta_i)\}_{i=1}^D$ is a set of parameterised unitaries, and D is the depth of circuit. We also assume that $U_i(\theta_i) = e^{-i\theta_i V_i}$ are rotations through angles θ_i, generated by Hermitian operators V_i with eigenvalues ± 1. In this case, one can compute partial derivatives of $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ using the parameter-shift rule [71], which reads

$$\partial_{\theta_i} q_{\boldsymbol{\theta}}(\boldsymbol{x}) = q_{\boldsymbol{\theta}_i^+}(\boldsymbol{x}) - q_{\boldsymbol{\theta}_i^-}(\boldsymbol{x}), \tag{3}$$

where $\boldsymbol{\theta}_{i^\pm} = \boldsymbol{\theta} \pm \frac{\pi}{4}\boldsymbol{e}_i$, with $\boldsymbol{e}_i$ a unit vector in the ith direction. More generally, this formula allows one to express the first-order partial derivative of an expectation of a function h as

$$\partial_{\theta_i} \mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}}(\boldsymbol{x})}[h(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}_i^+}(\boldsymbol{x})}[h(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}_i^+}(\boldsymbol{x})}[h(\boldsymbol{x})]. \tag{4}$$

The major challenge in using any implicit generative model is designing a suitable objective function. As noted before, one cannot compute $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ directly, and thus valid objectives can only involve statistical quantities (e.g., expectations) which can be efficiently computed using samples. For generative models based on QCBMs, various objectives have been proposed, including moment-matching, maximum mean discrepancy, Stein and Sinkhorn divergences, and adversarial objectives based on the Kullback–Leibler divergence. In this paper, we propose a more general class of objective functions—f-divergences—for training QCBMs.

2.3. Adversarial Generative Modelling with f-Divergences

Let $f : (0,\infty) \to \mathbb{R}$ be a convex function with $f(1) = 0$ and strict convexity at 1. Suppose that $p(\boldsymbol{x}) = 0$ whenever $q_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0$. The f-divergence, or Csiszár divergence [72,73], between $q_{\boldsymbol{\theta}}$ and p is defined as

$$D_f(p\|q_{\boldsymbol{\theta}}) = \mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}}(\boldsymbol{x})}\left[f\left(\frac{p(\boldsymbol{x})}{q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right)\right]. \tag{5}$$

Suppose instead that $q_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0$ whenever $p(\boldsymbol{x}) = 0$. Then the f-divergence can be written in terms of f^*, the convex conjugate of f, as

$$D_f(p\|q_{\boldsymbol{\theta}}) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[f^*\left(\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right)\right]. \tag{6}$$

In what follows, we will generally prefer this formulation, as it leads to simpler expressions.

The function f is called the *generator* of the divergence. For different choices of f, one obtains well-known divergences such as TV, KL, and χ^2. In this paper, we investigate the effect of this choice on the training of a QCBM. To ensure a fair comparison, we assume that the generators are standardised and normalised such that $f'(1) = 0$ and $f''(1) = 1$ [74]. This ensures that $D_f(p\|q_{\boldsymbol{\theta}}) \geq 0$ with equality if and only if $p \equiv q_{\boldsymbol{\theta}}$, even if p and $q_{\boldsymbol{\theta}}$ are unnormalised. Note that one can normalise and standardise

We minimise the f-divergence using gradient-based methods. We thus require the derivative of D_f with respect to $\boldsymbol{\theta}_i$. Using the chain and the parameter-shift rules, it is straightforward to compute

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_i} D_f(p\|q_{\boldsymbol{\theta}}) &= \sum_{\boldsymbol{x}} p(\boldsymbol{x})\partial_{\boldsymbol{\theta}_i} f^*\left(\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right) && (7)\\
&= \sum_{\boldsymbol{x}} p(\boldsymbol{x}) f^{*\prime}\left(\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right)\frac{1}{p(\boldsymbol{x})}\partial_{\boldsymbol{\theta}_i} q_{\boldsymbol{\theta}}(\boldsymbol{x}) && (8)\\
&= \sum_{\boldsymbol{x}} f^{*\prime}\left(\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right)\left(q_{\boldsymbol{\theta}_i^+}(\boldsymbol{x}) - q_{\boldsymbol{\theta}_i^-}(\boldsymbol{x})\right) && (9)\\
&= \mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}_i^+}(\boldsymbol{x})}\left[f^{*\prime}\left(\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right)\right] - \mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}_i^-}(\boldsymbol{x})}\left[f^{*\prime}\left(\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right)\right]. && (10)
\end{aligned}$$

We summarise some well-known f-divergences, the convex conjugates of their generators, and their parameter-shift rules, in Tables 1 and 2. We also plot some of the convex conjugate generators in Figure 1.

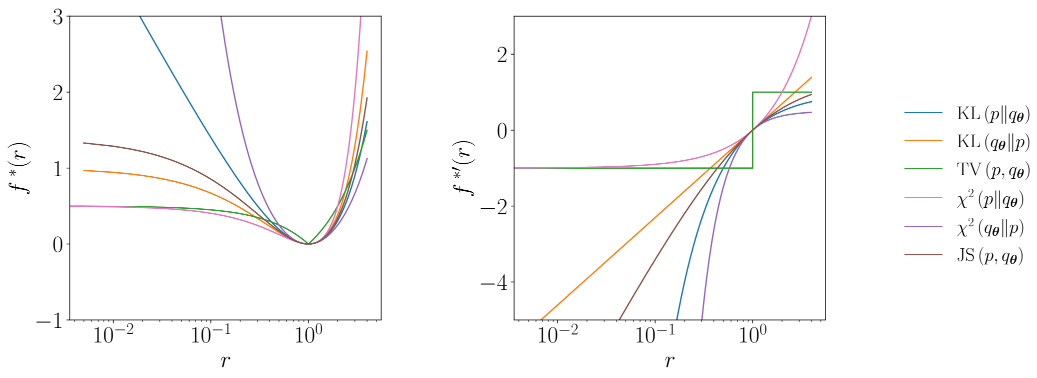


Figure 1. Convex conjugate f^* (left panel) and derivative $f^{*\prime}$ (right panel) of the generator f for several f-divergences. All generators have been standardised with $f'(1) = 0$ and normalised with $f''(1) = 1$, except for the TV.

Returning to Equation (10), it is clear that the problem of computing the gradient reduces to that of estimating the probability ratio $r(\boldsymbol{x}) = \frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}$. We choose to define $r(\boldsymbol{x})$ in this way since it is more natural when one is interested with writing the f-divergence in terms of f^*, as we do here. Note that in some literature the ratio is defined in the reverse manner by switching the probabilities. We can estimate the probability ratio from the output of a binary classifier [35]. Suppose we assign samples $\boldsymbol{x}\sim q_{\boldsymbol{\theta}}(\boldsymbol{x})$ to one class, and samples $\boldsymbol{x}{\sim}p(\boldsymbol{x})$ to another class. Suppose, in addition, that one has access to an exact binary classifier $d_*(\boldsymbol{x})$, which outputs the probability that the sample $\boldsymbol{x}$ originated from $q_{\boldsymbol{\theta}}(\boldsymbol{x})$. Then, assuming uniform prior probabilities for the two classes, it is straightforward to show via Bayes' theorem that (see Section 2.2 in [50]).

Table 1. A summary of well-known f-divergences, including the definition, the convex conjugate of the generator f^*, and the corresponding parameter-shift rule in terms of the ratio $r(\boldsymbol{x}) = \frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}$. The $\Vert$ symbol indicates that the divergence is asymmetric, while a comma indicates that it is symmetric. Interestingly, one can construct symmetric f-divergences for every asymmetric one (see Table 2).

f-Divergence	Definition	f^*	Parameter-Shift
total variation	$\mathrm{TV}(p,q_{\boldsymbol{\theta}}) = \frac{1}{2}\sum \lvert p(\boldsymbol{x}) - q_{\boldsymbol{\theta}}(\boldsymbol{x})\rvert$	$\frac{1}{2}\lvert r-1\rvert$	$\frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}+}}[\mathrm{sgn}(r(\boldsymbol{x})-1)] - \frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}-}}[\mathrm{sgn}(r(\boldsymbol{x})-1)]$
squared Hellinger	$\mathrm{H}^2(p,q_{\boldsymbol{\theta}}) = \sum\left(\sqrt{p(\boldsymbol{x})} - \sqrt{q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right)^2$	$2(\sqrt{r}-1)^2$	$-2\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\frac{1}{\sqrt{r(\boldsymbol{x})}}\right] + 2\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\frac{1}{\sqrt{r(\boldsymbol{x})}}\right]$
Kullback–Leibler (type I, forward)	$\mathrm{KL}(p\Vert q_{\boldsymbol{\theta}}) = \mathbb{E}_p\left[\log\frac{p(\boldsymbol{x})}{q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right]$	$-\log r + r - 1$	$-\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\frac{1}{r(\boldsymbol{x})}\right] + \mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\frac{1}{r(\boldsymbol{x})}\right]$
Kullback–Leibler (type I, reverse)	$\mathrm{KL}(q_{\boldsymbol{\theta}}\Vert p) = \mathbb{E}_{q_{\boldsymbol{\theta}}}\left[\log\frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}\right]$	$r\log r - r + 1$	$\mathbb{E}_{q_{\boldsymbol{\theta}+}}[\log r(\boldsymbol{x})] - \mathbb{E}_{q_{\boldsymbol{\theta}-}}[\log r(\boldsymbol{x})]$
Kullback–Leibler (type II, forward)	$\mathrm{KL}(p\Vert\frac{p+q_{\boldsymbol{\theta}}}{2}) = \mathbb{E}_p\left[\log\frac{2p(\boldsymbol{x})}{p(\boldsymbol{x})+q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right]$	$4\log\frac{2}{r+1} + 2(r-1)$	$-4\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\frac{1}{r(\boldsymbol{x})+1}\right] + 4\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\frac{1}{r(\boldsymbol{x})+1}\right]$
Kullback–Leibler (type II, reverse)	$\mathrm{KL}(q_{\boldsymbol{\theta}}\Vert\frac{p+q_{\boldsymbol{\theta}}}{2}) = \mathbb{E}_{q_{\boldsymbol{\theta}}}\left[\log\frac{2q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})+q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right]$	$4r\log\frac{2r}{r+1} + 2(1-r)$	$4\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\log\frac{r(\boldsymbol{x})}{r(\boldsymbol{x})+1} + \frac{1}{r(\boldsymbol{x})+1}\right] -4\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\log\frac{r(\boldsymbol{x})}{r(\boldsymbol{x})+1} + \frac{1}{r(\boldsymbol{x})+1}\right]$
Pearson (forward)	$\chi^2(p\Vert q_{\boldsymbol{\theta}}) = \sum\frac{(p(\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{x}))^2}{p(\boldsymbol{x})}$	$\frac{(r-1)^2}{2}$	$\mathbb{E}_{q_{\boldsymbol{\theta}+}}[r(\boldsymbol{x})] - \mathbb{E}_{q_{\boldsymbol{\theta}-}}[r(\boldsymbol{x})]$
Pearson (reverse)	$\chi^2(q_{\boldsymbol{\theta}}\Vert p) = \sum\frac{(p(\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{x}))^2}{q_{\boldsymbol{\theta}}(\boldsymbol{x})}$	$\frac{(r-1)^2}{2r}$	$-\frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\frac{1}{r(\boldsymbol{x})^2}\right] + \frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\frac{1}{r(\boldsymbol{x})^2}\right]$

Table 2. A summary of the symmetric f-divergences corresponding to some well-known asymmetric f-divergences, including the definition, and the parameter-shift rule.

f-Divergence	Definition	Parameter-Shift
symmetric Kullback–Leibler (type I, Jeffrey)	$\mathrm{J}(p,q_{\boldsymbol{\theta}}) = \mathrm{KL}(p\Vert q_{\boldsymbol{\theta}}) + \mathrm{KL}(q_{\boldsymbol{\theta}}\Vert p)$	$\frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\log r(\boldsymbol{x}) - \frac{1}{r(\boldsymbol{x})}\right] - \frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\log r(\boldsymbol{x}) - \frac{1}{r(\boldsymbol{x})}\right]$
symmetric Kullback–Leibler (type II, Jensen–Shannon)	$\mathrm{JS}(p,q_{\boldsymbol{\theta}}) = \mathrm{KL}(p\Vert\frac{p+q_{\boldsymbol{\theta}}}{2}) + \mathrm{KL}(q_{\boldsymbol{\theta}}\Vert\frac{p+q_{\boldsymbol{\theta}}}{2})$	$2\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[\log\frac{r(\boldsymbol{x})}{1+r(\boldsymbol{x})}\right] - 2\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[\log\frac{r(\boldsymbol{x})}{1+r(\boldsymbol{x})}\right]$
symmetric Pearson	$\bar{\chi}^2(p,q_{\boldsymbol{\theta}}) = \chi^2(p\Vert q_{\boldsymbol{\theta}}) + \chi^2(q_{\boldsymbol{\theta}}\Vert p)$	$\frac{1}{4}\mathbb{E}_{q_{\boldsymbol{\theta}+}}\left[2r(\boldsymbol{x}) - \frac{1}{r(\boldsymbol{x})^2}\right] - \frac{1}{4}\mathbb{E}_{q_{\boldsymbol{\theta}-}}\left[2r(\boldsymbol{x}) - \frac{1}{r(\boldsymbol{x})^2}\right]$

$$r(\boldsymbol{x}) = \frac{d_*(\boldsymbol{x})}{1-d_*(\boldsymbol{x})}. \tag{11}$$

In practice, we do not have access to the exact classifier $d_*(\boldsymbol{x})$. However, under the assumption that we can efficiently sample from both distributions, we can train a classifier $d_{\boldsymbol{\phi}}(\boldsymbol{x})$, parameterised by $\boldsymbol{\phi}$, to distinguish between the two distributions. One can use any proper scoring rule to train the classifier [50]. A typical choice is the negative cross entropy, given by

$$\mathcal{L}(\boldsymbol{\phi};\boldsymbol{\theta}) = -\mathbb{E}_{\boldsymbol{x}\sim q_{\boldsymbol{\theta}}(\boldsymbol{x})}\left[\log d_{\boldsymbol{\phi}}(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\log(1-d_{\boldsymbol{\phi}}(\boldsymbol{x}))\right]. \tag{12}$$

The classifier seeks to minimise this objective, which corresponds to low classification errors. We emphasise that, in this objective, $\boldsymbol{\theta}$ is fixed at the current QCBM parameters. The resulting classifier approximates the probability ratio for the current QCBM as

$$r(\boldsymbol{x}) \approx \frac{d_{\boldsymbol{\phi}}(\boldsymbol{x})}{1 - d_{\boldsymbol{\phi}}(\boldsymbol{x})}. \tag{13}$$

This can be plugged into Equation (10) to approximate the gradient. With this in mind, we define the cost function for the QCBM as

$$\mathcal{J}(\boldsymbol{\theta}; \boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{x} \sim q_{\boldsymbol{\theta}}(\boldsymbol{x})}\left[f^{*\prime}\left(\frac{d_{\boldsymbol{\phi}}(\boldsymbol{x})}{1 - d_{\boldsymbol{\phi}}(\boldsymbol{x})} \right) \right], \tag{14}$$

where now the parameters of the classifier $\boldsymbol{\phi}$ are fixed and the argument of the expectation value is independent of $\boldsymbol{\theta}$. The adversarial generative modelling can be regarded as the following optimisation problem

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}; \boldsymbol{\phi}), \tag{15}$$

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}; \boldsymbol{\theta}), \tag{16}$$

where the required expectation values are estimated from samples. In principle, the classifier can be trained to optimality in order to provide the best possible ratio for the generative model. Alternatively, the two objective functions can be optimised in tandem, using alternating gradient descent steps or a two-timescale gradient descent scheme [75].

3. Training Heuristics

3.1. Switching f-Divergences

In this Section, we describe a heuristic for dynamically switching between f-divergences throughout the training process of our generative model (specifically the QCBM).

To motivate this heuristic, we examine how $D_f(p \| q_{\boldsymbol{\theta}})$ varies with respect to values of $r(\boldsymbol{x}) = \frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})}$. We begin by noting that all f-divergences which can be standardised agree on the divergence between nearby distributions [76], but can otherwise exhibit very different behaviours. In particular, we focus on their initial rates of convergence.

One may rationalise the different rates of convergence for each divergence at the beginning of training by considering the following argument [50,64,77]. Consider n qubits, such that there are 2^n different values of $r(\boldsymbol{x})$. For a successful training, all these values need to converge towards 1 (which implies our goal that $q_{\boldsymbol{\theta}} \equiv p$). Now suppose we were to estimate the divergence in Equation (6) using a set of samples from the target distribution $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x})$. At the beginning of training, $q_{\boldsymbol{\theta}}$ is initialised at random and is therefore expected to be far from the target. This means that $q_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \ll p(\boldsymbol{x}_i)$ for most of the samples. In other words, at the beginning of training most of the samples yield probability ratios $r(\boldsymbol{x}_i) \ll 1$.

It is evident from the left panel of Figure 1 that some divergences, including TV, vary slowly in the region where $r \ll 1$, and are therefore more liable to saturation in the initial stages of training. Other divergences, such as forward KL and reverse χ^2, generate strong gradients in this region. In the limiting case where p and $q_{\boldsymbol{\theta}}$ have disjoint supports, TV and JS saturate, whereas forward KL diverges [78]. This problem is well known within the context of training generative adversarial networks; since an idealised formulation optimises JS, several alternative cost functions have been proposed to mitigate its slow initial convergence [64,77–79].

Though we can only apply this logic to the particular regime where p and $q_{\boldsymbol{\theta}}$ are far apart, it is also evident from Figure 1 that the f-divergences exhibit a wide diversity of behaviours throughout most of training. We propose to exploit this with the following heuristic. At every optimisation step, we choose an f-divergence for each direction in parameter space that generates the highest gradient in said direction. This requires no additional quantum circuit evaluations since we only need to evaluate Equation (10) for the different generators. Concretely, the heuristic can be written as follows. For each step, to update parameter $\boldsymbol{\theta}_i$, we choose the f-divergence labelled j, D_{f_j}, which obeys:

$$|\partial_{\boldsymbol{\theta}_i} D_{f_j}| > |\partial_{\boldsymbol{\theta}_i} D_{f_k}| \qquad \forall k \in \mathcal{F}. \tag{17}$$

For simplicity, in this paper, we restrict the set $\mathcal{F}$ to only contain those f-divergences illustrated in Figure 1. We call this heuristic *f-switch*.

3.2. Local Cost Functions

In this Section, we outline an alternative heuristic for training the QCBM, based on introducing locality into the cost function. Let us briefly provide some motivation for this approach. One of the most fundamental challenges associated with hybrid quantum-classical algorithms is the *barren plateau* phenomenon, whereby the gradient of the cost function vanishes exponentially in the number of qubits [36,37,80–88]. This phenomenon can arise due to deep unstructured ansätze [80], large entanglement [83,84], high levels of noise [88], and global cost functions [36,37]. As such, it is a rather general phenomenon in many quantum machine learning applications, including generative models. In the presence of barren plateaus, exponential precision (i.e., an exponential number of samples) is required in order to resolve against finite sampling noise and determine a minimising direction in the cost function landscape. Since the standard objective of quantum algorithms is to achieve a polynomial scaling in the system size (as opposed to the exponential scaling of classical algorithms), barren plateaus can destroy any hope of a variational quantum algorithm achieving quantum advantage.

Although, in this paper, we do not directly analyse the emergence of barren plateaus in the QCBM, we are nonetheless motivated by existing results on barren plateaus. We focus, in particular, on the connection between barren plateaus and global cost functions (i.e., cost functions defined in terms of global observables), given that such cost functions naturally arise in hybrid quantum-classical generative models. The connection between trainability and locality was first established by Cerezo et al. [36], who proved that cost functions defined in terms of global observables exhibit barren plateaus for *all* circuit depths in circuits composed of random two-qubit gates which act on alternating pairs of qubits (i.e., blocks forming local 2-designs). Meanwhile, local cost functions do not exhibit barren plateaus for shallow circuits; in this case, cost function gradients vanish at worst polynomially in the number of qubits.

On the basis of this result, there is clear motivation to seek a local cost function (i.e., a cost function defined in terms of local observables) for the hybrid quantum-classical generative model introduced in Section 2.3. We now attempt to make some progress towards this goal.

We write $q^i_{\boldsymbol{\theta}}(x_i)$ to denote the marginal distribution of the i^{th} element of the bit-string $\boldsymbol{x} = (x_1, \ldots, x_n)$. Using Jensen's inequality on Equation (6), it can be shown that the f-divergence between joint distributions is larger than the f-divergence between marginal distributions. Thus, we have

$$D_f(p(\boldsymbol{x}) \| q_{\boldsymbol{\theta}}(\boldsymbol{x})) \geq \frac{1}{n} \sum_{i=1}^{n} D_f(p^i(x_i) \| q^i_{\boldsymbol{\theta}}(x_i)). \tag{18}$$

Our heuristic consists of minimising the right-hand side of this inequality. Even though this is a lower-bound to the original cost, it is a fully local cost function. Later, we show how to generalise this approach allowing for a trade off between trainability and accuracy. We call this heuristic *f-local*.

Let us show the difference between the global cost function (left-hand side of the inequality) and the local cost function (right-hand side) by means of an example. For ease of exposition, we assume in this discussion that the f-divergence of interest is the reverse KL with generator $f^*(r) = r \log r - r + 1$. We emphasise, however, that the methodology is generic to any f-divergence. We begin by rewriting the expression in Equation (1) as

$$q_{\boldsymbol{\theta}}(\boldsymbol{x}) = \langle \mathbf{0}|U^\dagger(\boldsymbol{\theta})H_{\boldsymbol{x}}U(\boldsymbol{\theta})|\mathbf{0}\rangle, \tag{19}$$

where we have defined $H_{\boldsymbol{x}} := |\boldsymbol{x}\rangle\langle\boldsymbol{x}|$. We can thus write the reverse KL in the form of a generic cost function (see, e.g., [3]) as

$$\mathrm{KL}(q_{\boldsymbol{\theta}}\|p) = \sum_{\boldsymbol{x}} q_{\boldsymbol{\theta}}(\boldsymbol{x}) \log \frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p(\boldsymbol{x})} = \sum_{\boldsymbol{x}} g_{\boldsymbol{x}}\Big(\langle \mathbf{0}|U^\dagger(\boldsymbol{\theta})H_{\boldsymbol{x}}U(\boldsymbol{\theta})|\mathbf{0}\rangle\Big), \tag{20}$$

where we define $g_{\boldsymbol{x}}(q_{\boldsymbol{\theta}}) := q_{\boldsymbol{\theta}} \log \frac{q_{\boldsymbol{\theta}}}{p(\boldsymbol{x})}$. This cost function is clearly global, since the observables, $H_{\boldsymbol{x}}$, act on all qubits.

Now, rewriting Equation (20) in terms of the adversarial approximation in Equation (14), we have

$$\mathcal{J}(\boldsymbol{\theta};\boldsymbol{\phi}) = \sum_{\boldsymbol{x}} q_{\boldsymbol{\theta}}(\boldsymbol{x}) \,\mathrm{logit}(d_{\boldsymbol{\phi}}(\boldsymbol{x})) = \sum_{\boldsymbol{x}} h_{\boldsymbol{x}}\Big(\langle \mathbf{0}|U^\dagger(\boldsymbol{\theta})H_{\boldsymbol{x}}U(\boldsymbol{\theta})|\mathbf{0}\rangle\Big), \tag{21}$$

where $h_{\boldsymbol{x}}(q_{\boldsymbol{\theta}}) := q_{\boldsymbol{\theta}}\,\mathrm{logit}(d_{\boldsymbol{\phi}}(\boldsymbol{x}))$, and $\mathrm{logit}(y) := \log \frac{y}{1-y}$. It is interesting to note that the global observable $H_{\boldsymbol{x}}$ only enters into $h_{\boldsymbol{x}}(q_{\boldsymbol{\theta}})$ via the first term, namely $q_{\boldsymbol{\theta}}(\boldsymbol{x})$. It is arguable, however, that the second term in $h_{\boldsymbol{x}}(q_{\boldsymbol{\theta}})$, namely $\mathrm{logit}(d_{\boldsymbol{\phi}}(\boldsymbol{x}))$ should also be regarded as a global quantity.

We now consider the fully local cost function in the right-hand side of Equation (18). Applying the adversarial approximation to each of the n probability ratios, the QCBM objective is

$$\mathcal{J}^L(\boldsymbol{\theta};\boldsymbol{\phi}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{x_i\in\{0,1\}} q^i_{\boldsymbol{\theta}}(x_i)\,\mathrm{logit}\Big(d^i_{\boldsymbol{\phi}}(x_i)\Big) = \frac{1}{n}\sum_{i=1}^{n}\sum_{x_i\in\{0,1\}} h^L_{x_i}\Big(\langle \mathbf{0}|U^\dagger(\boldsymbol{\theta})H^L_{x_i}U(\boldsymbol{\theta})|\mathbf{0}\rangle\Big), \tag{22}$$

where we have replaced the global observable $H_{\boldsymbol{x}}$ in Equation (21) by the set of local observables

$$H^L_{x_i} = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|_i \otimes \mathbb{1}_{\bar{i}}. \tag{23}$$

Here, $|\boldsymbol{x}\rangle\langle\boldsymbol{x}|_i$ is a projector on the computational basis for qubit i, and $\mathbb{1}_{\bar{i}}$ denotes the identity on all qubits except qubit i. We have also replaced the 'global' function $h_{\boldsymbol{x}}(q_{\boldsymbol{\theta}})$ in Equation (21) by the set of local functions

$$h^L_{x_i}(p^i_{\boldsymbol{\theta}}) = q^i_{\boldsymbol{\theta}}\,\mathrm{logit}\Big(d^i_{\boldsymbol{\phi}}(x_i)\Big). \tag{24}$$

Here, $\{d^i_{\boldsymbol{\phi}}(x_i)\}_{i=1}^n$ is a set of n 'local' classifiers, which act only on the marginal distribution corresponding to the ith qubit. That is to say, $d^i_{\boldsymbol{\phi}}$ are trained to distinguish between samples $x_i \sim q^i_{\boldsymbol{\theta}}(x_i)$ and samples $x_i \sim p^i(x_i)$. One may ask why it is not sufficient to simply make only the observable, $H_{\boldsymbol{x}}$, local as is done in other literature addressing the barren plateau problem [36]. In our case, it turns out that if one does not *also* make the functions $h_{\boldsymbol{x}}$ local, in other words by keeping the classifier 'global', the cost function becomes intractable to compute due to a need to explicitly compute joint probabilities from the circuit, $q_{\boldsymbol{\theta}}$. This hints at the subtlety that appears when attempting to address barren plateaus in generative modelling, that does not necessarily exist in other variational algorithms.

We are, of course, interested in whether the local cost function is faithful to the original cost function. Recall that we are minimising the lower bound in Equation (18). It is clear that, if the local cost function is minimised, so that $D_f(q^i_{\boldsymbol{\theta}}\|p^i) = 0$ for all $i \in \{1,\ldots,n\}$, and all of the marginals coincide, there is still no guarantee that the joint distributions will be identical. This observation suggests that, while this cost function may be more trainable than the original cost function on account of its locality, it may also be significantly less accurate. In an attempt to remedy this, we can instead consider a more general k-local cost function which acts on subsets of k qubits. In particular, by defining $\boldsymbol{x}_{i:j} := (\boldsymbol{x}_i, \ldots, \boldsymbol{x}_j)$, we can introduce

$$\mathcal{J}^{L(k)}(\boldsymbol{\theta};\boldsymbol{\phi}) = \frac{1}{n-k+1}\sum_{i=1}^{n-k+1}\sum_{\boldsymbol{x}_{i:i+k-1}\in\{0,1\}^k} q_{\boldsymbol{\theta}}^{i:i+k-1}(\boldsymbol{x}_{i:i+k-1})\,\mathrm{logit}\Big(d_{\boldsymbol{\phi}}^{i:i+k-1}(\boldsymbol{x}_{i:i+k-1})\Big) \tag{25}$$

$$= \frac{1}{n-k+1}\sum_{i=1}^{n-k+1}\sum_{\boldsymbol{x}_{i:i+k-1}\in\{0,1\}^k} h^{L(k)}_{\boldsymbol{x}_{i:i+k-1}}\Big(\langle\mathbf{0}|U^{\dagger}(\boldsymbol{\theta})H^{L(k)}_{\boldsymbol{x}_{i:i+k-1}}U(\boldsymbol{\theta})|\mathbf{0}\rangle\Big), \tag{26}$$

where

$$H^{L(k)}_{\boldsymbol{x}_{i:i+k-1}} = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|_{i:i+k-1}\otimes\mathbb{1}_{\widetilde{i:i+k-1}}, \tag{27}$$

$$h^{L(k)}_{\boldsymbol{x}_{i:i+k-1}}(q_{\boldsymbol{\theta}}^{i:i+k-1}) = q_{\boldsymbol{\theta}}^{i:i+k-1}\,\mathrm{logit}\Big(d_{\boldsymbol{\phi}}^{i:i+k-1}(\boldsymbol{x}_{i:i+k-1})\Big), \tag{28}$$

and where $\{d_{\boldsymbol{\phi}}^{i:i+k-1}(\boldsymbol{x}_{i:i+k-1})\}_{i=1}^{n-k+1}$ is a set of $n-k+1$ 'k-local' classifiers, defined in an obvious fashion. This k-local cost function now approximates the sum of the reverse KL between the k-marginals (of neighbouring qubits) of the target distribution $p(\boldsymbol{x})$, and the variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{x})$.

Arguing as before, it is clear that the k-local cost function will admit additional global minima in comparison to the global cost function for any $1 \leq k < n$. In particular, when the k-local cost function is minimised, the k-nearest neighbour marginals of $p(\boldsymbol{x})$ and $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ coincide. One can expect, however, that as the value of k is increased, not only will the number of additional minima decrease, but the disparity between the joint distributions of the target and the model at these global minima will decrease. This suggests that in order to achieve a 'sweet spot' between trainability and accuracy, a reasonably approach is to start by optimising the k-local cost function with a small value of k (promoting trainability), before iteratively increasing the value of k (promoting accuracy) until $k = n$, thus recovering the global cost function.

We should remark that, while for ease of notation we have defined the k-local cost function in terms of marginals with respect to neighbouring qubits $(\boldsymbol{x}_i, \ldots, \boldsymbol{x}_{i+k-1})$, one can in theory choose any sets of qubits of size at most k (e.g., nearest neighbours, all possible combinations, and randomly sampled). In general, for a fixed value of k, this choice will influence the accuracy of the objective function, as well as its computational cost, and should be made on a case-by-case basis on the basis of the available computational resources.

4. Numerical Results

In this Section, we present numerical results to illustrate the performance of the training heuristics proposed in Section 3.

Preliminaries

Throughout this Section, we utilise a QCBM composed of alternating layers of single qubit gates and entangling gates (see Figure 2). We implement the quantum circuit using `pytket` [89] and execute the simulations with Qiskit [90]. The parameters of the QCBM are updated using stochastic gradient descent with a constant learning rate, which is tuned to each of the simulations.

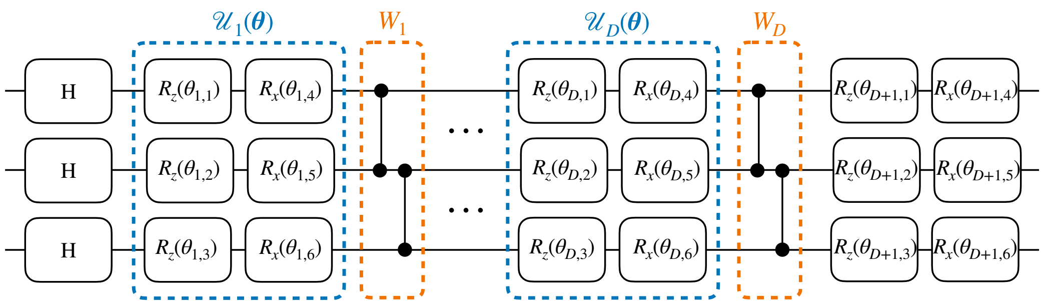
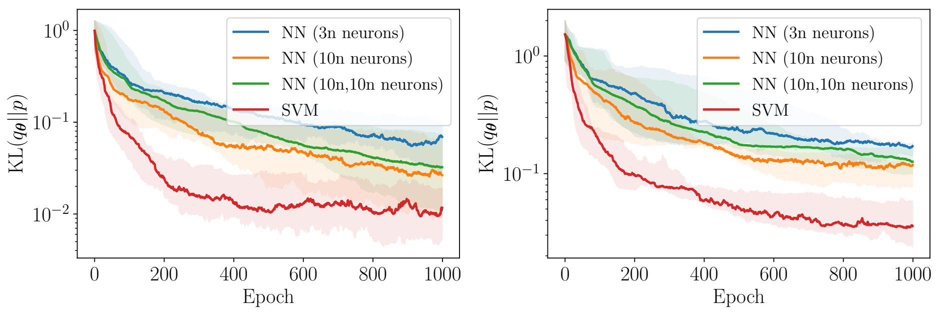


Figure 2. The ansatz employed in numerical simulations (shown for three qubits). The ansatz consists of D alternating layers of single qubit gates and entangling gates. The single qubit layers consists of two single qubit rotations, one around the z axis and one around the x axis. The entangling layer is composed of a ladder of CZ gates. There is an additional layer of Hadamard gates prior to the first layer, and an additional layer of single qubit rotations after the final layer. The total number of parameters in a circuit of depth D is given by $n_p = n(2D+2)$, where n is the number of qubits.

Regarding the classical component of the adversarial generative model (i.e., the binary classifier), we use either a fully connected feed-forward neural network with ReLU neurons (NN), or a support vector machine with RBF kernel (SVM). Indeed, one rather surprising byproduct of our numerical investigation is that the training performance of the adversarial generative model could be improved, at times significantly, by using a SVM in place of a NN for this component (see Figure 3). This, in itself, should be of some interest to practitioners. Not only can SVMs be faster to train, but they depend on significantly fewer hyper-parameters than NNs, whose performance is often highly dependent on careful tuning of the number of hidden layers, the number of neurons in each hidden layer, the learning rate, the batch size, etc. While we do not suggest that SVMs will always outperform NNs in this setting, this does indicate that SVMs may represent a viable alternative. We implement the NNs using PyTorch [91], while the SVMs are implemented with `scikit-learn` [92]. The particular hyper-parameters used in each simulation are specified below.



(a) 3 qubits **(b)** 4 qubits

Figure 3. Training performance of the QCBM in illustrative 3 qubit and 4 qubit experiments using 4 different classifiers. The classifiers are trained using 500 samples. We plot the bootstrapped median (solid line), as well as 90% confidence intervals (shaded).

In the majority of our numerical simulations, we consider a QCBM with 3 qubits. This corresponds to a discrete target distribution p which takes 2^3 values. We generally also assume that the target distribution corresponds to a particular instantiation of the QCBM, for a fixed number of layers, D_p. By varying the number of layers, D_{q_θ} used to train the generative model, we can then investigate different parameterisation regimes of interest. In the case that the number of layers used to generate the target is greater than the number of layers used in the model ($D_p > D_{q_{\boldsymbol{\theta}}}$), the model is under-parameterised (or severely under-parameterised). Meanwhile, when the number of layers used to generate the target and the number of layers used in the model are equal ($D_p = D_{q_{\boldsymbol{\theta}}}$), the model is said to be exactly parameterised. In these cases, a solution to the learning problem is guaranteed to exist: there exists $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ such that $p \equiv q_{\boldsymbol{\theta}_0}$. Finally, when the number of layers used to generate the target is less than the number used in the model ($D_p < D_{q_{\boldsymbol{\theta}}}$), the model is over-parameterised (or severely over-parameterised). We provide a more precise definition of these different cases, as applied to our numerics, in Table 3.

Table 3. The different parameterisation regimes used in the 3 qubit numerical simulations.

	Severely over Parameterised (OO)	**over Parameterised (O)**	**Exactly Parameterised (E)**	**under Parameterised (U)**	**Severely under Parameterised (UU)**
Number of parameters (layers) used to generate the target p	12 parameters (1 layer)	12 parameters (1 layer)	12 parameters (1 layer)	30 parameters (4 layers)	30 parameters (4 layers)
Number of parameters (layers) used for the model $q_{\boldsymbol{\theta}}$	30 parameters (4 layers)	24 parameters (3 layers)	12 parameters (1 layer)	18 parameters (2 layers)	12 parameters (1 layer)

For each of the settings (i.e., choice of circuit depth for the target and model, choice of heuristic, number of qubits) explored, we train the generative model using nine independent parameter initialisations. We then use a bootstrapping procedure to provide a more robust estimate of the median cost at each training epoch. We first take samples of size nine from the outcome of the nine independent experiments, 10,000 times with replacement. We then compute the median cost across each set of samples to obtain a distribution of 10,000 medians. Using this distribution, we compute the median and obtain error bars from the 5th and 95th percentiles, corresponding to a 90% confidence interval.

4.1. Switching f-Divergences

We begin by considering the performance of the heuristic introduced in Section 3.1. The f-divergences that can be standardised locally behave as KL to second order [76]. Notably, TV cannot be standardised; indeed, it is straightforward to show that TV provides an upper bound for all other f-divergences with $f''(1) = 1$ in this regime. For this reason, we evaluate both the exact TV and the exact KL to measure performance.

We begin by reporting the results obtained using an exact classifier, for each of the parameterisation regimes given in Table 3. The generator is trained using 1000 samples per iteration. The results are given in Table 4.

Table 4. Performance of the QCBM trained using the TV and the f-divergence heuristic for 3 qubits in over-, under-, and exactly parameterised regimes. We show the bootstrapped median of the TV (top two rows) and the KL (bottom two rows) after 500 epochs. The asterisk (*) on some of the experiments indicates that the cost is still converging. The bold indicates the regimes where f-switch significantly outperforms the other methods.

D_f **Evaluated**	D_f **Used in Training**	**OO (12, 30)**	**O (12, 24)**	**E (12, 12)**	**U (30, 18)**	**UU (30, 12)**
TV	TV	$(1.12^{+0.45}_{-0.28}) \times 10^{-2}$	$(8.4^{+1.2}_{-1.0}) \times 10^{-3}$	$(1.00^{+1.51}_{-0.12}) \times 10^{-2}$	$(1.06^{+0.26}_{-0.23}) \times 10^{-2}$	$(1.4^{+2.4}_{-0.7}) \times 10^{-2}$
TV	f-switch	$\boldsymbol{(0.6^{+3.8}_{-0.5}) \times 10^{-5}}$ *	$\boldsymbol{(2.5^{+2.5}_{-2.1}) \times 10^{-3}}$ *	$(3.1^{+1.8}_{-1.9}) \times 10^{-2}$	$(0.65^{+0.27}_{-0.51}) \times 10^{-2}$	$(1.8^{+2.9}_{-0.9}) \times 10^{-2}$
KL	TV	$(3.5^{+2.1}_{-1.3}) \times 10^{-4}$	$(2.0^{+0.6}_{-0.4}) \times 10^{-4}$	$(2.6^{+14.8}_{-2.3}) \times 10^{-3}$	$(3.7^{+1.7}_{-92.6}) \times 10^{-4}$	$(0.6^{+24.3}_{-0.4}) \times 10^{-3}$
KL	f-switch	$\boldsymbol{(0.0182^{+1.383}_{-0.012}) \times 10^{-8}}$	$(1.8^{+20.9}_{-1.7}) \times 10^{-5}$ *	$(3.5^{+9.1}_{-2.0}) \times 10^{-3}$	$(2.4^{+1.6}_{-2.4}) \times 10^{-4}$	$(1.8^{+4.3}_{-1.5}) \times 10^{-3}$

Our results indicate that the heuristic is able to outperform TV when the QCBM is (severely) over-parameterised. This may be due to the extra degrees of freedom in the model. These allow for more discrepancies between the loss landscapes of the f-divergences, which the heuristic is able to exploit. In Figures 4 and 5, we provide a more detailed illustration of the training performance of the f-switch heuristic in this regime. Figure 4 corresponds to an exact classifier: in this case, use of the heuristic significantly improves the convergence of the QCBM. Figure 5 corresponds to a trained classifier, trained on 1000 samples per iteration: in this case, use of the heuristic can lead to marginal performance improvements with respect to TV (left-hand figure). The remaining results in this Section are all reported for an exact classifier.

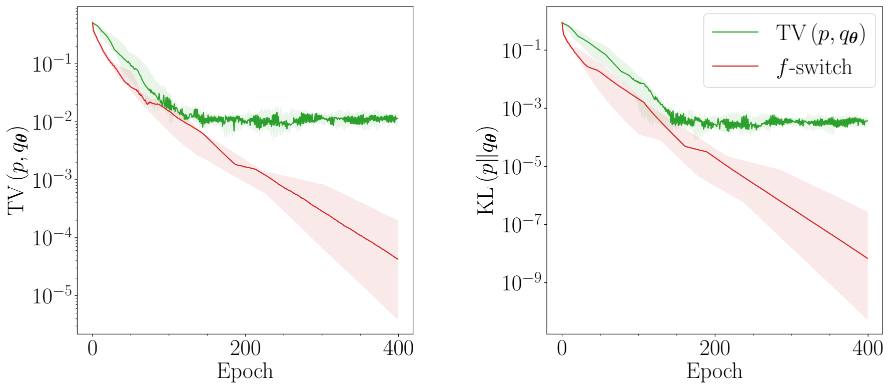


Figure 4. Performance of the QCBM trained using the TV (green) and the f-divergence heuristic (red) for 3 qubits in the severely over-parameterised case OO(12,30), using an exact classifier. We show the bootstrapped median (solid line) and 90% confidence intervals (shaded) of the TV (**left**) and the KL (**right**).

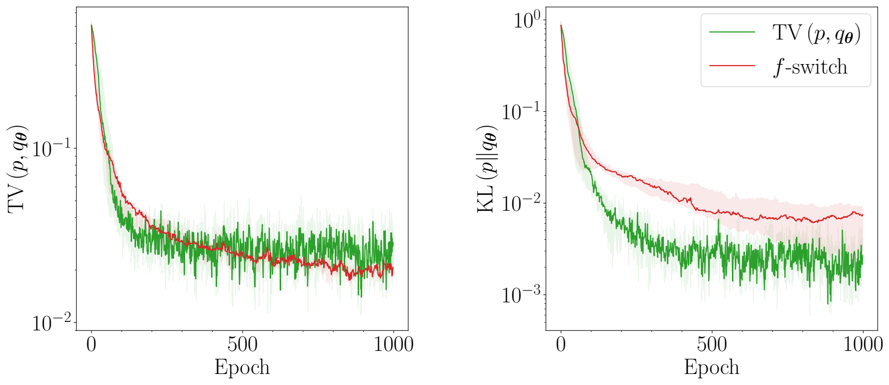


Figure 5. Performance of the QCBM training using the TV (green) and the f-divergence heuristic (red) for 3 qubits in the severely over-parameterised case OO(12,30), using a trained SVM classifier. We show the bootstrapped median (solid line) and 90% confidence intervals (shaded) of both the TV (**left**) and the KL (**right**).

The average performance of the heuristic is similar to TV in the exactly and under-parametrised regimes. There are, however, initial parameter configurations within these regimes for which the heuristic significantly outperforms TV. In Figure 6, we plot the median losses obtained throughout the training of the QCBM in the under-parametrised U(30, 18) regime. The best-performing experiment in this regime is also presented in Figure 7, alongside all the other f-divergences considered in Figure 1. After 200 epochs, the training method that solely uses TV has converged, but all the other divergences, including the heuristic, continue to converge exponentially quickly to smaller losses. In the under-parameterised regime, the ansatz is not guaranteed to contain the true solution. However, after reaching a KL of $\sim 10^{-3}$, these f-divergences traverse similar landscapes. Since the f-switch heuristic is shown to reach a KL of $\sim 10^{-5}$, we can assume that all of these f-divergences will converge to the global minimum, with the heuristic arriving first.

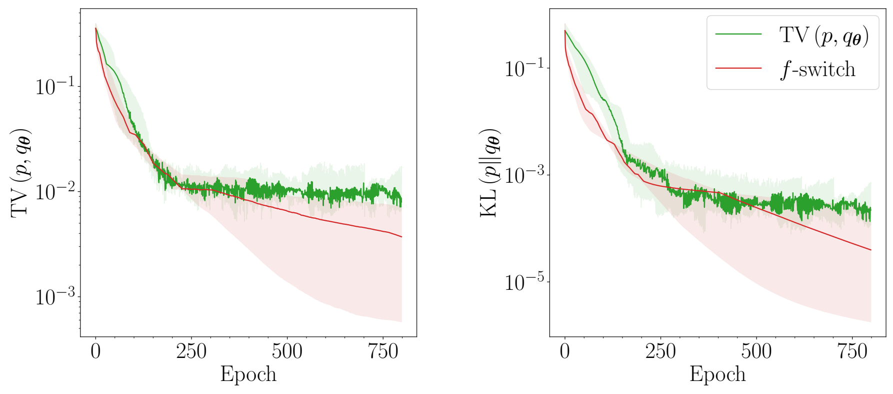


Figure 6. Performance of the QCBM training using the TV (green) and the f-divergence heuristic (red) for 3 qubits in the under-parameterised case U(30,18). We show the bootstrapped median (solid line) and 90% confidence intervals (shaded) of both the TV (**left**) and the KL (**right**).

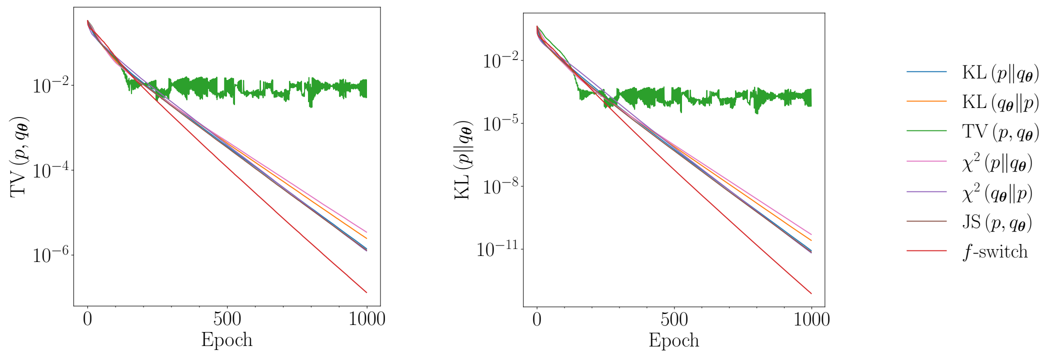


Figure 7. Performance of the QCBM trained using several f-divergences for 3 qubits in the under-parameterised case U(30, 18). The parameters are initialised using the parameters which gave the lowest cost during training in Figure 6. We show the exact TV (**left**) and the exact KL (**right**).

Finally, in Figure 8, we illustrate the mechanics of the f-switch heuristic. In particular, we plot which f-divergence is 'activated' for each direction in the parameter space, at each epoch of the training in Figure 7.

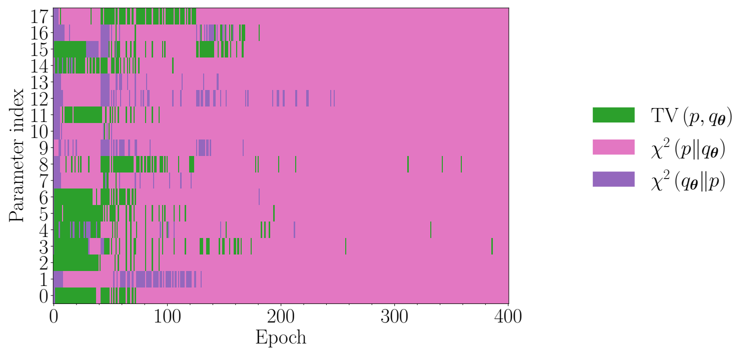
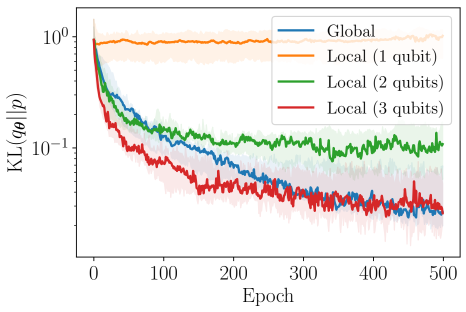


Figure 8. f-divergences chosen throughout the training of the heuristic in Figure 7 in each of the 18 directions in parameter space.

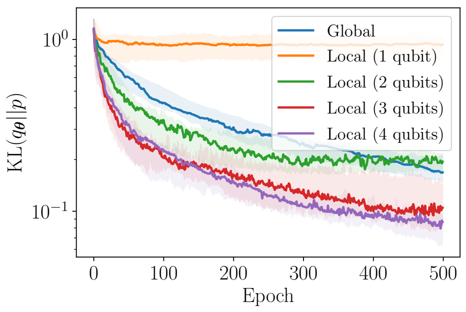
We remark that as the number of qubits is increased, the randomly initialised model and the target distributions are expected to be increasingly further apart. The heuristic can pick the divergence that provides the highest initial learning signal. For this reason, we expect the heuristic to become particularly useful as the number of qubits is increased.

4.2. Local Cost Functions

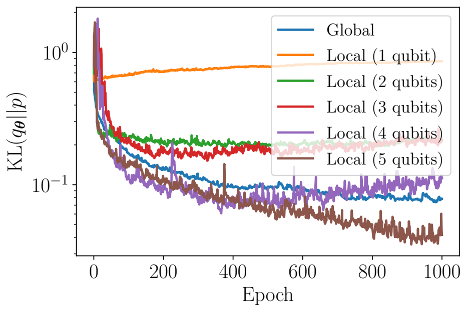
We now turn our attention to the heuristic introduced in Section 3.2, incorporating locality in the cost function, dubbed f-local. In this Section, the target distribution is a discretised Gaussian. All classifiers are neural networks with 1 hidden layer made of 10 k ReLU neurons, where k is the locality parameter. The number of layers in the QCBM equals the number of qubits, $D = n$. All expectation values are estimated using 500 samples. In Figure 9, we plot the training performance of the QCBM using the global cost function and several k-local cost functions, for $n = 4, 5$, and 6 qubit experiments. For 4 and 5 qubits, we show the bootstrapped median for the first 500 training epochs, as well as 90% confidence intervals. For 6 qubits, we plot an illustrative training example for the first 1000 training epochs.



(a) 4 qubits



(b) 5 qubits



(c) 6 qubits

Figure 9. Training performance of the QCBM using the global and local reverse KL for 4 qubits, 5 qubits, and 6 qubits, for a discretised Gaussian target distribution. For 4 qubits and 5 qubits, we show the bootstrapped median (solid line), as well as 90% confidence intervals (shaded). For 6 qubits, we plot an illustrative training example.

Let us make several remarks. Firstly, it would appear that the use of a k-local cost function can indeed improve the convergence (rate) of the training procedure, particularly during the initial stages. This improvement is increasingly evident as the number of qubits is increased. As such, this approach could be regarded as a potential strategy for tackling barren plateaus in higher-dimensional problems. However, we leave a thorough study of this phenomenon to future work.

Secondly, it is clear that the use of any k-local cost function will eventually prohibit convergence to the true target distribution. As discussed in Section 3.2, the k-local cost function is minimised whenever the k-marginal distributions of the target and the model coincide, which does not necessarily imply that their joint distributions are equal. The smaller the value of k, the greater the possible disparity between two distributions whose k-marginals coincide. This is clearly visualised in Figure 9: as the value of k decreases, the asymptotic reverse KL achieved during training with the k-local cost function plateaus at increasingly larger values.

As remarked previously, this suggests that an optimal training strategy may be to start the training procedure with a small value of k, before iteratively increasing the value of k as training proceeds. For example, let us consider the 5 qubit experiment in Figure 9b. Initially, the 3-local cost function (red) appears to yields the greatest convergence rate. After approximately 150 epochs, the 4-local cost function (purple) now seems to be favourable. Asymptotically, one can imagine that the global cost function (blue) will be preferable. One observes similar behaviour in the 6 qubit experiment in Figure 9c.

In practice, of course, it is not possible to compute the reverse KL directly, and thus another tractable metric is required in order to determine the optimal moment for switching between the k-local cost functions. Alternatively, one can simply increase the locality of the cost function after a set number of epochs.

5. Estimation of f-Divergences on Fault-Tolerant Quantum Computers

The above discussion is purely heuristic in nature and suitable for near-term quantum computers, but we can also address f-divergences from the other end of the spectrum; using fault-tolerant devices. In particular, we can leverage a recent line of study into quantum property testing of distributions. The key question here is whether or not a particular probability distribution has a certain property.

The work of [38] was one of the first to provide such an answer, demonstrating a quadratic speedup for determining whether two distributions over $[n]$ were close or ε-far in TV. These quantum algorithms typically work in the *oracle* model, and we measure run time relative to the number of queries to such an oracle (query complexity). In the classical case, we define oracle access to a distribution over $[n]$, $p = \{p_i\}_{i=1}^n$ as $O_p : [S] \to [n], S \in \mathbb{N}$. The oracle is a mechanism to translate a uniform distribution over $[S]$ to the true distribution over $[n]$. In the quantum case, such an oracle is replaced by a unitary operator, $\hat{O}_p$ acting on a state encoding $s \in [S]$, along with an ancillary register to ensure reversibility and defined as: $\hat{O}_p|s\rangle|0\rangle = |s\rangle|O_p(s)\rangle \forall s \in [S]$.

We begin our discussion with the TV. The authors of [38] produced a quantum property testing algorithm for the TV via an algorithm which actually *estimates* the TV quadratically faster. The analysis in [38] resulted in an algorithm to estimate the TV up to additive error ε, with probability of success of $1-\delta$, using $O(\sqrt{n}/\varepsilon^8\delta^5)$ samples. This was later improved by [39] to the following

Theorem 1 (Section 4, Montanaro [39])**.** *Assume p, q are two distributions on $[n]$. Then there is a quantum algorithm that approximates $TV(p,q)$ up to an additive error $\varepsilon > 0$, with probability of success $1-\delta$, using $O(\sqrt{n}\varepsilon^{-3/2}/\log(1/\delta))$ quantum queries.*

These ideas were extended in [40] to also give an algorithm for computing the (forward) KL quadratically faster than possibly classically (and also computing certain entropies of distributions). Due to the existence of the ratio p_i/q_i in the expression for the KL, we must make a further assumption, which was not necessary in the case of the TV distance in Theorem 1. This assumption will also be necessary when considering many of the other divergences in Table 1. In particular, we must assume the two distributions are such that: $p_i/q_i \leq g(n)$, $\forall i \in [n]$, for some $g : \mathbb{N} \to \mathbb{R}^+$. (This assumption is appropriate when one defines the KL in terms of the generator f and the ratio $r = p/q$. Conversely, when one defines the KL in terms of the convex conjugate f^* and the ratio $r = q/p$, then the appropriate assumption would instead be that $q_i/p_i \leq g(n), \forall i \in [n]$.) This assumption is also necessary in the classical case. With this, we then have

Theorem 2 (Theorem 4.1, Li and Wu [40])**.** *Assume p, q are two distributions on $[n]$ satisfying $p_i/q_i \leq g(n)$, $\forall i \in [n]$ for some $a : \mathbb{N} \to \mathbb{R}^+$. Then there is a quantum algorithm that approximates $KL(p\|q)$ within an additive error $\varepsilon > 0$ with probability of success at least $2/3$ using $\widetilde{O}(\sqrt{n}/\varepsilon^2)$ quantum queries to p and $\widetilde{O}(\sqrt{n}\,g(n)/\varepsilon^2)$ quantum queries to q. (The notation $\widetilde{O}(\cdot)$ ignores factors that are polynomial in $\log n$ and $\log 1/\varepsilon$.)*

These results cover two of the f-divergences we use above (see Table 1). In particular, the latter algorithm provide a quantum speedup since it is known that one requires

$\Omega(n/\log(n)), \Omega(ng(n)/\log(n))$ *classical* queries to p and q respectively to estimate the KL [93]. On the other hand, we get a speedup for the former algorithm since it is known one requires $\Theta(n^{2/3}\varepsilon^{-4/3})$ [94] queries to test if two distributions are near or far in TV classically, which is an easier problem than estimating the metric directly.

The key idea behind both of these algorithms is to use a subroutine known as *quantum probability estimation* or *quantum counting*, which is adapted from *quantum amplitude estimation*. This provides a quadratic speedup in producing estimates $\tilde{p}_i, \tilde{q}_i$, of probabilities p_i, q_i from the distributions p, q, which are specified via a quantum oracle. Once the estimates of $\tilde{p}_i, \tilde{q}_i$ have been produced via the quantum subroutine, both of the above algorithms reduce to simple classical post-processing. This post processing involves constructing a random variable, y, whose expectation value gives exactly the divergence we require. For TV and KL estimation, this random variable is given by

$$y_i^{\mathrm{TV}} := \frac{|p_i - q_i|}{p_i + q_i}, \tag{29}$$

$$y_i^{\mathrm{KL}} := \log\frac{p_i}{q_i} = \log p_i - \log q_i. \tag{30}$$

By sampling this random variable according to another distribution $r := (r_i)_{i=1}^n$ (to be defined below), the quantity of interest is exactly given as an expectation value, namely

$$\sum_i r_i^{\mathrm{TV}} y_i^{\mathrm{TV}} = \mathbb{E}[y^{\mathrm{TV}}] = \mathrm{TV}(p,q), \tag{31}$$

$$\sum_i r_i^{\mathrm{KL}} y_i^{\mathrm{KL}} = \mathbb{E}[y^{\mathrm{KL}}] = \mathrm{KL}(p\|q). \tag{32}$$

One can check [38,40] that the suitable random variables are given by

$$r_i^{\mathrm{TV}} = \frac{1}{2}(p_i + q_i), \tag{33}$$

$$r_i^{\mathrm{KL}} = q_i. \tag{34}$$

Due to the probabilistic nature of quantum mechanics, one cannot obtain the *exact* values of the probabilities required to compute these expectation values. We must settle instead for *approximations* of p, q, namely $\tilde{p}, \tilde{q}$. These estimates are achieved using the *quantum approximate counting* lemma, which is an application of quantum amplitude estimation [95]. The work in [40] considered two versions of this algorithm, called EstAmp and EstAmp'. The only difference between these two algorithms is the behavior when one of the probabilities, q_i, is sufficiently close to zero. This is problematic in the case of the KL estimation (and indeed entropy estimation) in [40] since the relevant quantities diverge as $q_i \to 0$. The same is true in our case, as q_i^{-1} appears in many f-divergences.

Theorem 3 (Theorem 13, Brassard et al. [95] and Theorem 2.3, Li and Wu [40])**.** *For any $k, M \in \mathbb{N}$, there is a quantum algorithm (named EstAmp) with M queries to a boolean function, $\chi : [S] \to \{0,1\}$ that outputs $\tilde{a} = \sin^2(\frac{l\pi}{M})$ for some $l \in \{0,\ldots,M-1\}$ such that*

$$Pr\left[\tilde{a} = \sin^2\left(\frac{l\pi}{M}\right)\right] = \frac{\sin^2(M\Delta\pi)}{M^2\sin^2(\Delta\pi)} \leq \frac{1}{(2M\Delta)^2}, \tag{35}$$

where $\Delta = |\omega - l/M|$. This promises $|\tilde{a} - a| \leq 2\pi k\frac{\sqrt{a(1-a)}}{M} + k^2\frac{\pi^2}{M^2}$ with probability at least $8/\pi^2$ for $k = 1$ and with probability greater than $1 - \frac{1}{2(k=1)}$ for $k \geq 2$. If $a = 0$ then $\tilde{a} = 0$.

The modified algorithm (EstAmp') outputs $\sin^2(\frac{\pi}{2M})$ when EstAmp outputs 0, and outputs the same as EstAmp otherwise. Now that we have a mechanism for estimating the probabilities, we need a final ingredient, which is the generic speedup of Monte Carlo methods from [39]

Theorem 4 (Theorem 5, Montanaro [39])**.** *Let $\mathcal{A}$ be a quantum algorithm with output X such that Var*$[X] \leq \sigma^2$*. Then for ε where $0 < \varepsilon < 4\sigma$, by using $O((\sigma/\varepsilon)\log^{3/2}(\sigma/\varepsilon)\log\log(\sigma/\varepsilon))$ executions of $\mathcal{A}$ and $\mathcal{A}^{-1}$, Algorithm* 3 *in [39] outputs an estimate $\tilde{\mathbb{E}}[X]$ of $\mathbb{E}[X]$ such that*

$$Pr\left[|\tilde{\mathbb{E}}[X] - \mathbb{E}[X]| \geq \varepsilon\right] \leq 1/5. \tag{36}$$

Using these results, we now extend Theorems 1 and 2 to cover another f-divergence in Table 1: the forward Pearson divergence, $\chi^2(p\|q)$. The convex conjugate of the generator for this divergence is given by $f^*(r) = \frac{1}{2}(r-1)^2$ or, equivalently, $f^*(r) = \frac{1}{2}(r^2-1)$. The equivalence of these two generators is straightforward to demonstrate. In particular, we have

$$\mathbb{E}_p\left[\left(\frac{q_i}{p_i}-1\right)^2\right] = \sum_i p_i\left(\frac{q_i}{p_i}\right)^2 - 2\sum_i p_i\frac{q_i}{p_i} + \sum_i p_i = \sum_x p_i\left(\frac{q_i}{p_i}\right)^2 - 1 = \mathbb{E}_p\left[\left(\frac{q_i}{p_i}\right)^2 - 1\right]. \tag{37}$$

In fact, in what follows, we make use of the following representation:

$$\chi^2(p\|q) := \frac{1}{2}\sum_i p_i\left[\left(\frac{q_i}{p_i}\right)^2 - 1\right] = \sum_i q_i\left(\frac{q_i}{p_i}-1\right) := \sum_i r_i^{FP} y_i^{FP}, \tag{38}$$

where we have identified $r_i^{FP} = q_i$ and $y_i^{FP} = \frac{1}{2}(\frac{q_i}{p_i}-1)$. Using this representation, we develop the following Algorithm 1 for estimating the forward Pearson divergence.

Algorithm 1: Estimate the forward Pearson divergence of $p = (p_i)_{i=1}^n$ and $(q_i)_{i=1}^n$ on $[n]$.

```
Set: $l = \Omega\Big((\sigma/\varepsilon)\log^{3/2}(\sigma/\varepsilon)\log\log(\sigma/\varepsilon)\Big), \sigma := g(n)^2\left[1 + \frac{\exp\left(-\frac{\varepsilon^2}{2n}\right)}{g(n)^2}\right]$.
Set: The following subroutine to be the algorithm $\mathcal{A}$:
begin
    Sample an index, $i \in [n]$, according to p. Use the procedure EstAmp' with
      $2^{\lceil\log_2(\sqrt{n}g(n)/\varepsilon)\rceil}$ and $2^{\lceil\log_2(\sqrt{n}g(n)^2/\varepsilon)\rceil}$ queries to $q$ and $p$, respectively. Obtain
      estimates of $\tilde{p}_i, \tilde{q}_i$. Output $\tilde{y}_i^{\mathrm{FP}} = \frac{1}{2}\left(\frac{\tilde{q}_i}{\tilde{p}_i} - 1\right)$.
end
Use $\mathcal{A}$ for $l$ times in Theorem 4 to output an estimate, $\tilde{\chi}^2(p\|q)$ for $\chi^2(p\|q)$.
```

The query complexity of this Algorithm is contained in the following theorem. We defer the proof of this result, which is largely a technical extension of the proof(s) in [40], to Appendix A.

Theorem 5. *Assume p, q are two distributions on $[n]$ satisfying $q_i/p_i \leq g(n)$, $\forall i \in [n]$ for some $a : \mathbb{N} \to \mathbb{R}^+$. Then there is a quantum algorithm that approximates $\mathcal{X}^2(p\|q)$ within an additive error $\varepsilon > 0$ with probability of success at least $2/3$ using $\tilde{O}(\sqrt{n}g(n)/\varepsilon^2)$ quantum queries to q and $\tilde{O}(\sqrt{n}g(n)^2/\varepsilon^2)$ quantum queries to p.*

6. Discussion

Each f-divergence, with its unique operational meaning, finds application in information theory, statistics, and machine learning. In this paper, we showed that a generative model called quantum circuit born machine can be trained by efficiently minimising *any* f-divergence. The key observation is that a probabilistic classifier can be trained adversarially to provide an approximation to such divergences.

Building on this, we developed heuristics aimed at improving convergence of the generative training. The first heuristic, *f-switch*, lets each parameter minimise a different f-divergence. Numerical results with an ideal exact classifier show that this heuristic can converge faster and to better minima than when using a single f-divergence. However, in a more realistic setting where the classifier is trained adversarially, f-switch yields results similar to those obtained by minimising a single f-divergence.

The second training heuristic, *f-local*, consists of using a single f-divergence approximated by local cost functions. Numerical results show that, as the number of qubits increases, this strategy yield improved convergence of the generative training than when using a global cost function. To the best of our knowledge this is the first proposal of cost functions for generative modelling that can interpolate between trainability and accuracy. Extensive numerical simulations will be needed to confirm whether f-local can alleviate the barren plateau problem in generative modelling.

Interestingly, our local cost functions approximate the f-divergence using an ensemble of local binary classifiers. If the target probability distribution is known to have a particular conditional independence structure (e.g., it is defined by a Bayesian network or a Hidden Markov model), this information could be used to inform the choice of local classifiers.

One interesting research direction is to adapt the above heuristics to work with other families of distance measures. Of particular interest, integral probability metrics (IPMs) include the maximum mean discrepancy, the Dudley metric and the Wasserstein distance. While f-divergences are defined in terms of probability ratios, IPMs are defined in terms of probability differences. However, it is know that under suitable constraints margin-based classifiers yield estimators for IPMs [96]. This suggests that an extension of our heuristics to IPMs could be possible.

In this work, we also discussed the possibility of estimating certain f-divergences on a fault-tolerant quantum computer, therefore avoiding the use of classifiers. Previously published work has proven quadratic quantum speedups for the estimation of total variation [38,39] and forward Kullback–Leibler (KL) of type I [40]. Using these algorithms a quadratic speedups is achievable for the reverse KL of type I, and thus for the symmetric KL of type I (also known as Jeffrey divergence). It is plausible that with some refinements these algorithms can provide quadratic speedups for the KL of type II as well.

We contributed to this topic with an algorithm for estimating Pearson χ^2 divergences and by providing its query complexity. Interestingly, high-order Pearson divergences (also known as Vajda divergences) can be used to approximate any other f-divergence via Taylor expansion [97]. Generalising our quantum algorithm to Vajda divergences would therefore provide a way to estimate all other f-divergences on a fault-tolerant quantum computer.

Author Contributions: C.L. devised and implemented the f-switch heuristic. L.S. devised and implemented the f-local heuristic. B.C. and L.S. devised and analysed the fault-tolerant algorithm. C.L. and L.S. designed the figures. M.B. and B.C. supervised the work. All authors analysed the results and contributed to the final manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Cambridge Quantum Computing.

Data Availability Statement: Data used to generate the above figures are available upon request from the authors.

Acknowledgments: M.B. would like to thank Mattia Fiorentini for helpful conversations.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Proof of Theorem 5

In this Appendix, we provide a proof of Theorem 5. For completeness, we first repeat the theorem here.

Theorem 5. *Assume p, q are two distributions on $[n]$ satisfying $q_i/p_i \leq g(n)$, $\forall i \in [n]$ for some $a : \mathbb{N} \to \mathbb{R}^+$. Then there is a quantum algorithm that approximates $\mathcal{X}^2(p\|q)$ within an additive error $\varepsilon > 0$ with probability of success at least $2/3$ using $\tilde{O}(\sqrt{n}g(n)/\varepsilon^2)$ quantum queries to q and $\tilde{O}(\sqrt{n}g(n)^2/\varepsilon^2)$ quantum queries to p.*

Proof. We prove this theorem in two parts, following closely the approach in [40]. We are first required to show that the expectation of the output of the sub-routine $\mathcal{A}$, namely $\tilde{E} = \sum_{i\in[n]} q_i(\tilde{q}_i/\tilde{p}_i - 1)$ is sufficiently close to $E = \sum_{i\in[n]} q_i(q_i/p_i - 1)$. We begin by observing the following inequality. Let $x, y > 0$. In addition, suppose there exists $0 < K < \infty$ such that $y \leq K$. Then

$$|y - x| \leq K\left|\frac{y-x}{y}\right| = K\left|\frac{y/x - 1}{y/x}\right| \leq K\left|\log\left(\frac{y}{x}\right)\right| = K|\log(y) - \log(x)|. \tag{A1}$$

where we have used the elementary inequality: $(z-1)/z \leq \log(z)$. We then have, using the linearity of expectation,

$$|E - \tilde{E}| \leq \frac{1}{2}\sum_{i\in[n]} q_i \mathbb{E}\left[\left|\left(\frac{q_i}{p_i}\right) - \left(\frac{\tilde{q}_i}{\tilde{p}_i}\right)\right|\right] \tag{A2}$$

$$\leq \frac{1}{2}g(n)\sum_{i\in[n]} q_i \mathbb{E}\left[\left|\log\left(\frac{q_i}{p_i}\right) - \log\left(\frac{\tilde{q}_i}{\tilde{p}_i}\right)\right|\right]. \tag{A3}$$

The remainder of the proof follows [40], with the roles of p and q now reversed, and with an additional factor of $g(n)$. In particular, using elementary properties of the logarithm, we have

$$|E - \tilde{E}| \leq \frac{1}{2}g(n)\sum_{i\in[n]} q_i\mathbb{E}[|\log q_i - \log\tilde{q}_i|] + \frac{1}{2}g(n)\sum_{i\in[n]} q_i\mathbb{E}[|\log p_i - \log\tilde{p}_i|] \tag{A4}$$

$$\leq \frac{1}{2}g(n)\sum_{i\in[n]} q_i\mathbb{E}[|\log q_i - \log\tilde{q}_i|] + \frac{1}{2}g(n)^2\sum_{i\in[n]} p_i\mathbb{E}[|\log p_i - \log\tilde{p}_i|], \tag{A5}$$

where in the second line we have used the assumption that $q_i/p_i \leq g(n)$ for all $i \in [n]$. By (IV.5) and (IV.6) in ([40], Section IV), $2^{\lceil \log_2(\sqrt{n}g(n)/\varepsilon)\rceil}$ queries to q and $2^{\lceil \log_2(\sqrt{n}g(n)^2/\varepsilon)\rceil}$ queries to p yield

$$\sum_{i\in[n]} q_i\mathbb{E}[|\log q_i - \log\tilde{q}_i|] = \mathcal{O}\left(\frac{\varepsilon}{g(n)}\right), \tag{A6}$$

$$\sum_{i\in[n]} p_i\mathbb{E}[|\log p_i - \log\tilde{p}_i|] = \mathcal{O}\left(\frac{\varepsilon}{g(n)^2}\right). \tag{A7}$$

Substituting these bounds into Equation (A5), and re-scaling Algorithm 1 by a large enough constant, we obtain $|E - \tilde{E}| \leq \frac{\varepsilon}{2}$. We are now required to bound the variance of this random variable. The variance is at most

$$\frac{1}{4}\sum_{i\in[n]} q_i\left(\frac{\tilde{q}_i}{\tilde{p}_i} - 1\right)^2 = \frac{1}{4}\sum_{i\in[n]:\tilde{q}_i\leq\tilde{p}_i} q_i\left(\frac{\tilde{q}_i}{\tilde{p}_i} - 1\right)^2 + \frac{1}{4}\sum_{i\in[n]:\tilde{q}_i>\tilde{p}_i} q_i\left(\frac{\tilde{q}_i}{\tilde{p}_i} - 1\right)^2. \tag{A8}$$

We first turn our attention to the first term. Recall that EstAmp' outputs $\tilde{q}_i$ such that $\tilde{q}_i \geq \sin^2(\pi/2^{\lceil\log_2(\sqrt{n}g(n)/\varepsilon)\rceil+1}) \geq \varepsilon^2/(4ng(n)^2)$ for any i. It follows that $\tilde{q}_i/\tilde{p}_i \geq \tilde{q}_i \geq \varepsilon^2/(4ng(n)^2)$, and thus $\exp(-2\tilde{q}_i/\tilde{p}_i) \leq \exp\left(-\varepsilon^2/(2ng(n)^2)\right)$. We thus have, using also the fact that $(x-1)^2 < \exp(-2x)$ for $x > -1$, that

$$\frac{1}{4}\sum_{i:\tilde{q}_i<\tilde{p}_i} q_i\left(\frac{\tilde{q}_i}{\tilde{p}_i} - 1\right)^2 \leq \frac{1}{4}\sum_{i:\tilde{q}_i<\tilde{p}_i} q_i\exp\left(-2\frac{\tilde{q}_i}{\tilde{p}_i}\right) \tag{A9}$$

$$\leq \frac{1}{4}\sum_{i:\tilde{q}_i<\tilde{p}_i} q_i\exp\left(-\frac{\varepsilon^2}{2ng(n)^2}\right) \leq \exp\left(-\frac{\varepsilon^2}{2ng(n)^2}\right). \tag{A10}$$

Meanwhile, for the second term, using the fact that $(\tilde{q}_i/\tilde{p}_i - 1)^2 \leq [\tilde{q}_i/\tilde{p}_i]^2$ since, in this summation, $\tilde{q}_i/\tilde{p}_i \geq 1 \geq 1/2$, we obtain

$$\frac{1}{4}\sum_{i:\tilde{q}_i\geq\tilde{p}_i} q_i\left(\frac{\tilde{q}_i}{\tilde{p}_i}-1\right)^2 \leq \frac{1}{4}\sum_{i:\tilde{q}_i\geq\tilde{p}_i} q_i\left[\frac{\tilde{q}_i}{\tilde{p}_i}\right]^2 \leq \frac{1}{4}\sum_{i:\tilde{q}_i\geq\tilde{p}_i} q_i g(n)^2 \leq g(n)^2. \tag{A11}$$

Substituting Equations (A10) and (A11) into Equation (A8), we see that the variance of the random variable is at most

$$g(n)^2 + \exp\left(-\frac{\varepsilon^2}{2ng(n)^2}\right) = O\left(g(n)^2\left[1+\frac{\exp\left(-\frac{\varepsilon^2}{2ng(n)^2}\right)}{g(n)^2}\right]\right). \tag{A12}$$

It follows from Corollary 2 in [40] that we can approximate $\tilde{E}$ up to an additive error of $\varepsilon/2$ with probability of success of at least $2/3$ using $\widetilde{O}(1/\varepsilon)\cdot 2^{\lceil\log_2(\sqrt{n}g(n)/\varepsilon)\rceil} = \widetilde{O}(\sqrt{n}g(n)/\varepsilon^2)$ queries to q and $\widetilde{O}(1/\varepsilon)\cdot 2^{\lceil\log_2(\sqrt{n}g(n)^2/\varepsilon)\rceil} = \widetilde{O}(\sqrt{n}g(n)^2/\varepsilon^2)$ queries to p. Together with our earlier demonstration that $|E-\tilde{E}|\leq\varepsilon/2$, this completes the proof. □

References

1. McClean, J.R.; Romero, J.; Babbush, R.; Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.* **2016**, *18*, 23023. [CrossRef]
2. Benedetti, M.; Lloyd, E.; Sack, S.; Fiorentini, M. Parameterized quantum circuits as machine learning models. *Quantum Sci. Technol.* **2019**, *4*, 043001. [CrossRef]
3. Cerezo, M.; Arrasmith, A.; Babbush, R.; Benjamin, S.C.; Endo, S.; Fujii, K.; McClean, J.R.; Mitarai, K.; Yuan, X.; Cincio, L.; et al. Variational quantum algorithms. *Nat. Rev. Phys.* **2021**, *3*, 1–20. [CrossRef]
4. Bharti, K.; Cervera-Lierta, A.; Kyaw, T.H.; Haug, T.; Alperin-Lea, S.; Anand, A.; Degroote, M.; Heimonen, H.; Kottmann, J.S.; Menke, T.; et al. Noisy intermediate-scale quantum (NISQ) algorithms. *arXiv* **2021**, arXiv:2101.08448.
5. Li, W.; Deng, D.L. Recent advances for quantum classifiers. *arXiv* **2021**, arXiv:2108.13421.
6. Grant, E.; Benedetti, M.; Cao, S.; Hallam, A.; Lockhart, J.; Stojevic, V.; Green, A.G.; Severini, S. Hierarchical quantum classifiers. *NPJ Quantum Inf.* **2018**, *4*, 65. [CrossRef]
7. Cong, I.; Choi, S.; Lukin, M.D. Quantum convolutional neural networks. *Nat. Phys.* **2019**, *15*, 1273–1278. [CrossRef]
8. Schuld, M.; Killoran, N. Quantum Machine Learning in Feature Hilbert Spaces. *Phys. Rev. Lett.* **2019**, *122*, 40504. [CrossRef]
9. Havlíček, V.; Córcoles, A.D.; Temme, K.; Harrow, A.W.; Kandala, A.; Chow, J.M.; Gambetta, J.M. Supervised learning with quantum-enhanced feature spaces. *Nature* **2019**, *567*, 209–212. [CrossRef]
10. LaRose, R.; Coyle, B. Robust data encodings for quantum classifiers. *Phys. Rev. A* **2020**, *102*, 032420. [CrossRef]
11. Romero, J.; Olson, J.P.; Aspuru-Guzik, A. Quantum autoencoders for efficient compression of quantum data. *Quantum Sci. Technol.* **2017**, *2*, 45001. [CrossRef]
12. Pepper, A.; Tischler, N.; Pryde, G.J. Experimental Realization of a Quantum Autoencoder: The Compression of Qutrits via Machine Learning. *Phys. Rev. Lett.* **2019**, *122*, 60501. [CrossRef] [PubMed]
13. Ding, Y.; Lamata, L.; Sanz, M.; Chen, X.; Solano, E. Experimental Implementation of a Quantum Autoencoder via Quantum Adders. *Adv. Quantum Technol.* **2019**, *2*, 1800065. [CrossRef]
14. Otterbach, J.S.; Manenti, R.; Alidoust, N.; Bestwick, A.; Block, M.; Bloom, B.; Caldwell, S.; Didier, N.; Fried, E.S.; Hong, S.; et al. Unsupervised Machine Learning on a Hybrid Quantum Computer. *arXiv* **2017**, arXiv:1712.05771.
15. Liu, J.G.; Wang, L. Differentiable learning of quantum circuit Born machines. *Phys. Rev. A* **2018**, *98*, 62324. [CrossRef]
16. Benedetti, M.; Garcia-Pintos, D.; Perdomo, O.; Leyton-Ortega, V.; Nam, Y.; Perdomo-Ortiz, A. A generative modeling approach for benchmarking and training shallow quantum circuits. *NPJ Quantum Inf.* **2019**, *5*, 45. [CrossRef]
17. Hamilton, K.E.; Dumitrescu, E.F.; Pooser, R.C. Generative model benchmarks for superconducting qubits. *Phys. Rev. A* **2019**, *99*, 62323. [CrossRef]
18. Zhu, D.; Linke, N.M.; Benedetti, M.; Landsman, K.A.; Nguyen, N.H.; Alderete, C.H.; Perdomo-Ortiz, A.; Korda, N.; Garfoot, A.; Brecque, C.; et al. Training of quantum circuits on a hybrid quantum computer. *Sci. Adv.* **2019**, *5*, eaaw9918. [CrossRef]
19. Coyle, B.; Mills, D.; Danos, V.; Kashefi, E. The Born supremacy: Quantum advantage and training of an Ising Born machine. *NPJ Quantum Inf.* **2020**, *6*, 60. [CrossRef]
20. Du, Y.; Hsieh, M.H.; Liu, T.; Tao, D. Expressive power of parametrized quantum circuits. *Phys. Rev. Res.* **2020**, *2*, 33125. [CrossRef]
21. Anand, A.; Romero, J.; Degroote, M.; Aspuru-Guzik, A. Noise Robustness and Experimental Demonstration of a Quantum Generative Adversarial Network for Continuous Distributions. *Adv. Quantum Technol.* **2021**, *4*, 2000069. [CrossRef]
22. Leyton-Ortega, V.; Perdomo-Ortiz, A.; Perdomo, O. Robust implementation of generative modeling with parametrized quantum circuits. *Quantum Mach. Intell.* **2021**, *3*, 17. [CrossRef]
23. Dallaire-Demers, P.L.; Killoran, N. Quantum generative adversarial networks. *Phys. Rev. A* **2018**, *98*, 12324. [CrossRef]
24. Hu, L.; Wu, S.H.; Cai, W.; Ma, Y.; Mu, X.; Xu, Y.; Wang, H.; Song, Y.; Deng, D.L.; Zou, C.L.; et al. Quantum generative adversarial learning in a superconducting quantum circuit. *Sci. Adv.* **2019**, *5*, eaav2761. [CrossRef]

25. Zeng, J.; Wu, Y.; Liu, J.G.; Wang, L.; Hu, J. Learning and inference on generative adversarial quantum circuits. *Phys. Rev. A* **2019**, *99*, 52306. [CrossRef]
26. Zoufal, C.; Lucchi, A.; Woerner, S. Quantum Generative Adversarial Networks for learning and loading random distributions. *NPJ Quantum Inf.* **2019**, *5*, 103. [CrossRef]
27. Verdon, G.; Marks, J.; Nanda, S.; Leichenauer, S.; Hidary, J. Quantum Hamiltonian-Based Models and the Variational Quantum Thermalizer Algorithm. *arXiv* **2019**, arXiv:1910.02071.
28. Huang, H.L.; Du, Y.; Gong, M.; Zhao, Y.; Wu, Y.; Wang, C.; Li, S.; Liang, F.; Lin, J.; Xu, Y.; et al. Experimental Quantum Generative Adversarial Networks for Image Generation. *arXiv* **2020**, arXiv:2010.06201
29. Situ, H.; He, Z.; Wang, Y.; Li, L.; Zheng, S. Quantum generative adversarial network for generating discrete distribution. *Inf. Sci.* **2020**, *538*, 193–208. [CrossRef]
30. Coyle, B.; Henderson, M.; Le, J.C.J.; Kumar, N.; Paini, M.; Kashefi, E. Quantum versus classical generative modelling in finance. *Quantum Sci. Technol.* **2021**, *6*, 024013. [CrossRef]
31. Liu, W.; Zhang, Y.; Deng, Z.; Zhao, J.; Tong, L. A hybrid quantum-classical conditional generative adversarial network algorithm for human-centered paradigm in cloud. *EURASIP J. Wirel. Commun. Netw.* **2021**, *2021*, 37. [CrossRef]
32. Rudolph, M.S.; Toussaint, N.B.; Katabarwa, A.; Johri, S.; Peropadre, B.; Perdomo-Ortiz, A. Generation of High-Resolution Handwritten Digits with an Ion-Trap Quantum Computer. *arXiv* **2020**, arXiv:2012.03924.
33. Benedetti, M.; Coyle, B.; Fiorentini, M.; Lubasch, M.; Rosenkranz, M. Variational inference with a quantum computer. *arXiv* **2021**, arXiv:2103.06720.
34. Cheng, S.; Chen, J.; Wang, L. Information Perspective to Probabilistic Modeling: Boltzmann Machines versus Born Machines. *Entropy* **2018**, *20*, 583. [CrossRef] [PubMed]
35. Sugiyama, M.; Suzuki, T.; Kanamori, T. *Density Ratio Estimation in Machine Learning*; Cambridge University Press: Cambridge, UK, 2012.
36. Cerezo, M.; Sone, A.; Volkoff, T.; Cincio, L.; Coles, P.J. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nat. Commun.* **2021**, *12*, 1791. [CrossRef] [PubMed]
37. Uvarov, A.V.; Biamonte, J.D. On barren plateaus and cost function locality in variational quantum algorithms. *J. Phys. A Math. Theor.* **2021**, *54*, 245301. [CrossRef]
38. Bravyi, S.; Harrow, A.W.; Hassidim, A. Quantum Algorithms for Testing Properties of Distributions. *IEEE Trans. Inf. Theory* **2011**, *57*, 3971–3981. [CrossRef]
39. Montanaro, A. Quantum speedup of Monte Carlo methods. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2015**, *471*, 20150301. [CrossRef] [PubMed]
40. Li, T.; Wu, X. Quantum Query Complexity of Entropy Estimation. *IEEE Trans. Inf. Theory* **2019**, *65*, 2899–2921. [CrossRef]
41. Bowman, S.R.; Vilnis, L.; Vinyals, O.; Dai, A.M.; Jozefowicz, R.; Bengio, S. Generating Sentences from a Continuous Space. *arXiv* **2016**, arXiv:1511.06349.
42. Zhu, J.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2242–2251. [CrossRef]
43. Simonovsky, M.; Komodakis, N. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. In Proceedings of the 27th Int. Conf. Artif. Neural Networks, Rhodes, Greece, 4–7 October 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 412–422. [CrossRef]
44. Sinha, S.; Ebrahimi, S.; Darrell, T. Variational Adversarial Active Learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 5971–5980. [CrossRef]
45. Ha, D.; Schmidhuber, J. World Models. *arXiv* **2018**, arXiv:1803.10122.
46. Ilse, M.; Tomczak, J.M.; Louizos, C.; Welling, M. DIVA: Domain Invariant Variational Autoencoders. *arXiv* **2019**, arXiv:1905.10427.
47. Brehmer, J.; Kling, F.; Espejo, I.; Cranmer, K. MadMiner: Machine Learning-Based Inference for Particle Physics. *Comput. Softw. Big Sci.* **2020**, *4*, 3. [CrossRef]
48. Oord, A.V.D.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; Kavukcuoglu, K. WaveNet: A Generative Model for Raw Audio. *arXiv* **2016**, arXiv:1609.03499.
49. Diggle, P.J.; Gratton, R.J. Monte Carlo Methods of Inference for Implicit Statistical Models. *J. R. Stat. Soc. Ser. B* **1984**, *46*, 193–212. [CrossRef]
50. Mohamed, S.; Lakshminarayanan, B. Learning in Implicit Generative Models. *arXiv* **2017**, arXiv:1610.03483.
51. Frey, B.J. *Graphical Models for Machine Learning and Digital Communication*; MIT Press: Cambridge, MA, USA, 1998.
52. Uria, B.; Côté, M.A.; Gregor, K.; Murray, I.; Larochelle, H. Neural Autoregressive Distribution Estimation. *arXiv* **2016**, arXiv:1605.02226.
53. Rippel, O.; Adams, R.P. High-Dimensional Probability Estimation with Deep Density Models. *arXiv* **2013**, arXiv:1302.5125.
54. Rezende, D.J.; Mohamed, S. Variational Inference with Normalizing Flows. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1530–1538.
55. Dinh, L.; Sohl-Dickstein, J.; Bengio, S. *Density Estimation Using Real NVP*; ICLR: 2017. Available online: https://arxiv.org/abs/1605.08803 (accessed on 27 September 2021).
56. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2014**, arXiv:1312.6114.

57. Rezende, D.J.; Mohamed, S.; Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1278–1286.
58. Ackley, D.H.; Hinton, G.E.; Sejnowski, T.J. A learning algorithm for boltzmann machines. *Cogn. Sci.* **1985**, *9*, 147–169. [CrossRef]
59. Hinton, G.E.; Osindero, S.; Teh, Y. A fast learning algorithm for deep belief nets. *Neural Comput.* **2006**, *18*, 1527–1554. [CrossRef]
60. Salakhutdinov, R.; Hinton, G. Deep Boltzmann Machines. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Clearwater Beach, FL, USA, 16–18 April 2009; pp. 448–455.
61. Bengio, Y.; Thibodeau-Laufer, E.; Alain, G.; Yosinski, J. Deep Generative Stochastic Networks Trainable by Backprop. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014.
62. Dziugaite, G.K.; Roy, D.M.; Ghahramani, Z. Training generative neural networks via maximum mean discrepancy optimization. In Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence, Amsterdam, The Netherlands, 12–16 July 2015; pp. 258–267.
63. Li, Y.; Swersky, K.; Zemel, R. Generative Moment Matching Networks. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1718–1727.
64. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems—Volume 2, Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
65. Born, M. Zur Quantenmechanik der Stoßvorgänge. *Z. Phys.* **1926**, *37*, 863–867. [CrossRef]
66. Glasser, I.; Sweke, R.; Pancotti, N.; Eisert, J.; Cirac, J.I. Expressive power of tensor-network factorizations for probabilistic modeling. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019.
67. Bremner, M.J.; Montanaro, A.; Shepherd, D.J. Average-Case Complexity Versus Approximate Simulation of Commuting Quantum Computations. *Phys. Rev. Lett.* **2016**, *117*, 80501. [CrossRef]
68. Boixo, S.; Isakov, S.V.; Smelyanskiy, V.N.; Babbush, R.; Ding, N.; Jiang, Z.; Bremner, M.J.; Martinis, J.M.; Neven, H. Characterizing quantum supremacy in near-term devices. *Nat. Phys.* **2018**, *14*, 595–600. [CrossRef]
69. Bouland, A.; Fefferman, B.; Nirkhe, C.; Vazirani, U. On the complexity and verification of quantum random circuit sampling. *Nat. Phys.* **2019**, *15*, 159–163. [CrossRef]
70. Arute, F.; Arya, K.; Babbush, R.; Bacon, D.; Bardin, J.C.; Barends, R.; Biswas, R.; Boixo, S.; Brandao, F.G.S.L.; Buell, D.A.; et al. Quantum supremacy using a programmable superconducting processor. *Nature* **2019**, *574*, 505–510. [CrossRef] [PubMed]
71. Mitarai, K.; Negoro, M.; Kitagawa, M.; Fujii, K. Quantum circuit learning. *Phys. Rev. A* **2018**, *98*, 032309, doi:10.1103/PhysRevA.98.032309 [CrossRef]
72. Csiszár, I. Information-type measures of difference of probability distributions and indirect observation. *Stud. Sci. Math. Hung.* **1967**, *2*, 229–318.
73. Ali, S.M.; Silvey, S. A General Class of Coefficients of Divergence of One Distribution from Another. *J. R. Stat. Soc. Ser.-Methodol.* **1966**, *28*, 131–142. [CrossRef]
74. Amari, S. α-Divergence Is Unique, Belonging to Both f-Divergence and Bregman Divergence Classes. *IEEE Trans. Inf. Theory* **2009**, *55*, 4925–4931. [CrossRef]
75. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *arXiv* **2018**, arXiv:1706.08500.
76. Csiszár, I.; Shields, P. *Information Theory and Statistics: A Tutorial*; Foundations and Trends® in Communications and Information Theory; Now Publishers Inc.: Boston, MA, USA, 2004; Volume 1, pp. 417–528. [CrossRef]
77. Uehara, M.; Sato, I.; Suzuki, M.; Nakayama, K.; Matsuo, Y. Generative Adversarial Nets from a Density Ratio Estimation Perspective. *arXiv* **2016**, arXiv:1610.02920.
78. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein Generative Adversarial Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; Volume 70, pp. 214–223.
79. Nowozin, S.; Cseke, B.; Tomioka, R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. In Proceedings of the 30th Conference on Neural Information Processing Systems, Spain, Barcelona, 5–10 December 2016.
80. McClean, J.R.; Boixo, S.; Smelyanskiy, V.N.; Babbush, R.; Neven, H. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.* **2018**, *9*, 4812. [CrossRef]
81. Grant, E.; Wossnig, L.; Ostaszewski, M.; Benedetti, M. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum* **2019**, *3*, 214. [CrossRef]
82. Arrasmith, A.; Cerezo, M.; Czarnik, P.; Cincio, L.; Coles, P.J. Effect of barren plateaus on gradient-free optimization. *arXiv* **2020**, arXiv:2011.12245.
83. Marrero, C.O.; Kieferová, M.; Wiebe, N. Entanglement Induced Barren Plateaus. *arXiv* **2021**, arXiv:2010.15968.
84. Patti, T.L.; Najafi, K.; Gao, X.; Yelin, S.F. Entanglement devised barren plateau mitigation. *Phys. Rev. Res.* **2021**, *3*, 033090. [CrossRef]
85. Arrasmith, A.; Holmes, Z.; Cerezo, M.; Coles, P.J. Equivalence of quantum barren plateaus to cost concentration and narrow gorges. *arXiv* **2021**, arXiv:2104.05868.
86. Holmes, Z.; Sharma, K.; Cerezo, M.; Coles, P.J. Connecting ansatz expressibility to gradient magnitudes and barren plateaus. *arXiv* **2021**, arXiv:2101.02138.

87. Larocca, M.; Czarnik, P.; Sharma, K.; Muraleedharan, G.; Coles, P.J.; Cerezo, M. Diagnosing barren plateaus with tools from quantum optimal control. *arXiv* **2021**, arXiv:2105.14377.
88. Wang, S.; Fontana, E.; Cerezo, M.; Sharma, K.; Sone, A.; Cincio, L.; Coles, P.J. Noise-Induced Barren Plateaus in Variational Quantum Algorithms. *arXiv* **2021**, arXiv:2007.14384.
89. Sivarajah, S.; Dilkes, S.; Cowtan, A.; Simmons, W.; Edgington, A.; Duncan, R. t | ket>: A retargetable compiler for NISQ devices. *Quantum Sci. Technol.* **2020**, *6*, 14003. [CrossRef]
90. Aleksandrowicz, G.; Alexander, T.; Barkoutsos, P.; Bello, L.; Ben-Haim, Y.; Bucher, D.; Cabrera-Hernández, F.J.; Carballo-Franquis, J.; Chen A.; Chen, C.-F.; et al. Qiskit: An Open-source Framework for Quantum Computing. 2021. Available online: https://zenodo.org/record/2562111#.YVUWKzURXIU (accessed on 27 September 2021).
91. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv* **2019**, arXiv:1912.01703 [cs.LG].
92. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
93. Han, Y.; Jiao, J.; Weissman, T. Minimax Estimation of Divergences Between Discrete Distributions. *IEEE J. Sel. Areas Inf. Theory* **2020**, *1*, 814–823. [CrossRef]
94. Chan, S.O.; Diakonikolas, I.; Valiant, G.; Valiant, P. Optimal Algorithms for Testing Closeness of Discrete Distributions. *arXiv* **2013**, arXiv:1308.3946.
95. Brassard, G.; Høyer, P.; Mosca, M.; Tapp, A. Quantum amplitude amplification and estimation. *Quantum Comput. Inf.* **2002**, *305*, 53–74. [CrossRef]
96. Sriperumbudur, B.K.; Fukumizu, K.; Gretton, A.; Schölkopf, B.; Lanckriet, G.R.G. On integral probability metrics, ϕ-divergences and binary classification. *arXiv* **2009**, arXiv:0901.2698.
97. Nielsen, F.; Nock, R. On the chi square and higher-order chi distances for approximating f-divergences. *IEEE Signal Process. Lett.* **2014**, *21*, 10–13. [CrossRef]